

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
please do not report the images to the
Image Problems Mailbox.**

2/5/1 (Item 1 from file: 351)
DIALOG(R) File 351: Derwent WPI
(c) 2001 Derwent Info Ltd. All rts. reserv.

012130007 **Image available**

WPI Acc No: 1998-546919/199847

XRFX Acc No: N98-426149

Encryption algorithm development support apparatus for data communication between computers - includes encryption diagram execution environmental inspection unit for examining contents of block encryption diagram execution environmental storage unit

Patent Assignee: HITACHI SOFTWARE ENG CO LTD (HISF)

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
JP 10240511	A	19980911	JP 9741217	A	19970225	199847 B

Priority Applications (No Type Date): JP 9741217 A 19970225

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
JP 10240511	A	34	G06F-009/06	

Abstract (Basic): JP 10240511 A

The apparatus consists of an encryption diagram edit unit (201) to edit a diagrammatic expression of an encryption algorithm provided by a user based on algorithm specification stored beforehand. An encryption diagram storing unit (204) stores the edited diagram.

An encryption interpretation execution unit (202) interprets the stored diagram in the form of algorithm. An environmental storage unit (205) stores the situation of a variable used while interpreting the diagrammatic expression. An environmental inspection unit (203) examines the contents of the environment storage unit.

ADVANTAGE - Provides efficient decipherment technique. Raises efficiency in developing encryption program and block encryption algorithm. Offers efficient testing of program.

Dwg.7/30

Title Terms: ENCRYPTION; ALGORITHM; DEVELOP; SUPPORT; APPARATUS; DATA; COMMUNICATE; COMPUTER; ENCRYPTION; DIAGRAM; EXECUTE; ENVIRONMENT; INSPECT ; UNIT; CONTENT; BLOCK; ENCRYPTION; DIAGRAM; EXECUTE; ENVIRONMENT; STORAGE; UNIT

Derwent Class: P85; T01; W01

International Patent Class (Main): G06F-009/06

International Patent Class (Additional): G06F-011/28; G09C-001/00; H04L-009/06; H04L-009/10

File Segment: EPI; EngPI

2/5/2 (Item 1 from file: 347)
DIALOG(R) File 347: JAPIO
(c) 2000 JPO & JAPIO. All rts. reserv.

05957411 **Image available**

CODING ALGORITHM DEVELOPMENT ASSISTING DEVICE AND CODING PROGRAM DEVELOPMENT ASSISTING DEVICE

PUB. NO.: 10-240511 A]

PUBLISHED: September 11, 1998 (19980911)

INVENTOR(s): KUMETA AKIFUMI

KOMURO MUTSUMI

TSUTSUMI TOSHIYUKI

APPLICANT(s): HITACHI SOFTWARE ENG CO LTD [472485] (A Japanese Company or Corporation), JP (Japan)

APPL. NO.: 09-041217 [JP 9741217]

FILED: February 25, 1997 (19970225)

INTL CLASS: [6] G06F-009/06; G06F-011/28; G09C-001/00; G09C-001/00; H04L-009/06; H04L-009/10

JAPIO CLASS: 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units); 44.3 (COMMUNICATION -- Telegraphy); 44.9 (COMMUNICATION --

Other)

ABSTRACT

PROBLEM TO BE SOLVED: To efficiently assist the development of encryption algorithm which is adaptive to new decoding technology by assisting the description and editing of block encryption algorithm according to a predetermined block encryption algorithm scale.

SOLUTION: A user describes and edits block encryption algorithm diagram representation on the basis of the block encryption algorithm scale by using an encryption diagram editing device 201, and stores it in an encryption diagram storage device 204. To confirm that block encryption algorithm stored in the encryption diagram storage device 204 operates as the user intends, the block encryption algorithm is actually executed by using an encryption diagram interpretation execution device 202. This execution result is stored as a table in an encryption diagram execution environment storage device 205. Lastly, the user examines the execution result in detail by using an encryption diagram execution environment inspecting device 203.

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平10-240511

(43) 公開日 平成10年(1998) 9月11日

(51) Int.Cl. ⁶	識別記号	F I	
G 0 6 F 9/06	5 3 0	G 0 6 F 9/06	5 3 0 V
			5 3 0 H
11/28	3 4 0	11/28	3 4 0 A
G 0 9 C 1/00	6 1 0	G 0 9 C 1/00	6 1 0 A
	6 6 0		6 6 0 Z

審査請求 未請求 請求項の数 6 O L (全 34 頁) 最終頁に続く

(21) 出願番号 特願平9-41217

(22) 出願日 平成9年(1997) 2月25日

(71) 出願人 000233055

日立ソフトウェアエンジニアリング株式会
社

神奈川県横浜市中区尾上町6丁目81番地

(72) 発明者 久米田 暁文

神奈川県横浜市中区尾上町6丁目81番地

日立ソフトウェアエンジニアリング株式会
社内

(72) 発明者 小室 睦

神奈川県横浜市中区尾上町6丁目81番地

日立ソフトウェアエンジニアリング株式会
社内

(74) 代理人 弁理士 秋田 収喜

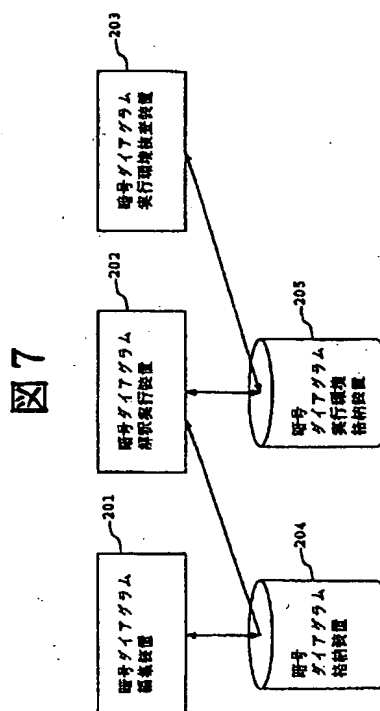
最終頁に続く

(54) 【発明の名称】 暗号アルゴリズム開発支援装置及び暗号プログラム開発支援装置

(57) 【要約】

【課題】 ブロック暗号アルゴリズムおよびブロック暗号プログラムの開発を効率良く支援可能にする。

【解決手段】 予め定義したブロック暗号アルゴリズム仕様記述記法によるブロック暗号アルゴリズムのダイアグラム表現をユーザが記述及び編集するのを支援する暗号ダイアグラム編集手段と、該暗号ダイアグラム編集手段によって編集されたブロック暗号アルゴリズムのダイアグラム表現を格納する暗号ダイアグラム格納手段と、格納されたブロック暗号アルゴリズムのダイアグラム表現を解釈実行する暗号ダイアグラム解釈実行手段と、解釈実行中のブロック暗号アルゴリズムのダイアグラム表現に現れる変数の状態を格納する暗号ダイアグラム実行環境格納手段と、このブロック暗号ダイアグラム実行環境格納手段の内容を検査するための暗号ダイアグラム実行環境検査手段と、を備える。



【特許請求の範囲】

【請求項1】 予め定義したブロック暗号アルゴリズム仕様記述記法によるブロック暗号アルゴリズムのダイアグラム表現をユーザが記述及び編集するのを支援する暗号ダイアグラム編集手段と、

該暗号ダイアグラム編集手段によって編集されたブロック暗号アルゴリズムのダイアグラム表現を格納する暗号ダイアグラム格納手段と、

格納されたブロック暗号アルゴリズムのダイアグラム表現を解釈実行する暗号ダイアグラム解釈実行手段と、

解釈実行中のブロック暗号アルゴリズムのダイアグラム表現に現れる変数の状態を格納する暗号ダイアグラム実行環境格納手段と、

このブロック暗号ダイアグラム実行環境格納手段の内容を検査するための暗号ダイアグラム実行環境検査手段と、を備えることを特徴とするブロック暗号アルゴリズム開発支援装置。

【請求項2】 ブロック暗号アルゴリズム仕様記述記法は、ビット列データを値とする変数のダイアグラム表現と、ビット列データ上の演算のダイアグラム表現とを構成要素として有し、前記変数と前記演算を組み合わせるためのダイアグラム表現として、前記変数からの参照操作のダイアグラム表現と、前記変数への代入操作のダイアグラム表現と、前記ビット列データへの前記演算の適用操作のダイアグラム表現と、前記ビット列データの分割操作のダイアグラム表現と、前記ビット列データの連結操作のダイアグラム表現と、前記ビット列データの複製操作のダイアグラム表現と、前記ビット列データへの繰り返し処理のダイアグラム表現と、前記ビット列データ上の演算を定義するためのダイアグラム表現とを備えることを特徴とする請求項1記載のブロック暗号アルゴリズム開発支援方法。

【請求項3】 テストケースを区別するためのIDと暗号化鍵と平文と該平文を前記暗号化鍵によって暗号化した暗号文からなるブロック暗号プログラムテストケースを格納するための暗号プログラムテストケース格納手段と、

前記IDを生成するためのID計数手段と、

前記暗号化鍵を生成するための鍵生成手段と、

前記平文を生成するための平文生成手段とを備え、

前記暗号文を生成するブロック暗号アルゴリズム開発支援手段と、を備えることを特徴とするブロック暗号プログラム開発支援装置。

【請求項4】 ブロック暗号アルゴリズムのダイアグラム表現の構文を解析するダイアグラム構文解析手段と、その解析結果であるダイアグラム構文木を格納するダイアグラム構文木格納手段と、

格納されたダイアグラム構文木からプログラム設計仕様を生成する設計仕様生成手段と、

ダイアグラムとプログラム設計仕様との対応関係を定

義した対応テーブルと、

生成された設計仕様を格納するための暗号プログラム設計仕様格納手段とを備えることを特徴とするブロック暗号プログラム開発支援装置。

【請求項5】 ブロック暗号プログラムの設計仕様の構文を解析する設計仕様構文解析手段と、

その解析結果である設計仕様構文木を格納する設計仕様構文木格納手段と、

格納された設計仕様構文木からプログラムを生成するプログラム生成手段と、

設計仕様とプログラムの間の対応関係を定義した対応テーブルと、

生成されたプログラムを格納するための暗号プログラム格納手段とを備えることを特徴とするブロック暗号プログラム開発支援装置。

【請求項6】 ブロック暗号プログラムのテストスケジュールを格納するテストスケジュール格納手段と、

このテストスケジュール格納手段に格納されたテストスケジュールに従ってテストケースを選択するテストケース選択手段と、

ブロック暗号プログラムを格納する暗号プログラム格納手段と、

格納されたブロック暗号プログラムに前記選択されたテストケースの構成要素である鍵と平文を与えて実行させるプログラム実行手段と、

その実行結果である暗号文と前記選択されたテストケースの構成要素である暗号文とを比較するための比較手段と、

前記選択されたテストケースのIDと該比較結果とからなるテスト結果を格納するための暗号プログラムテスト結果格納手段とを備えることを特徴とするブロック暗号プログラム開発支援装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、予め定義しておいたブロック暗号アルゴリズム仕様記述記法に基づいてブロック暗号アルゴリズムの開発を支援する装置および該アルゴリズムを実現するプログラムの開発を支援する装置に関するものである。

【0002】

【従来の技術】コンピュータネットワーク技術の発達等によって情報の共有化が進むに従い、情報の漏洩を防ぐための暗号技術に対する必要性が高まっている。暗号には大きく分けて、暗号化と復号化の時に同じ鍵を使用する秘密鍵暗号と、個人鍵と公開鍵という異なる鍵の組を使用する公開鍵暗号がある。秘密鍵暗号は、暗号通信を行なう際に鍵を送信者と受信者で共有する方法が問題となる。公開鍵暗号にはこの問題がない。

【0003】しかし、公開鍵暗号は秘密鍵暗号に比べて、暗号化と復号化に要する計算量が多い。このた

3

め、両者は用途によって使い分けられる。

【0004】秘密鍵暗号としてよく用いられているものに、固定長のデータ（ブロック）を単位として暗号化と復号化を行なうブロック暗号がある。ブロック暗号のアルゴリズムはこれまで種々のものが考案されている。文献「Federal Information Processing Standards Publication 46-2 1993 December 30; Announcing the DATA ENCRYPTION STANDARD (DES)」に紹介されているDESはその代表例である。

【0005】ところで、このようなブロック暗号アルゴリズムやそれを実現する手段に関する研究開発が進むにつれて、解読技術も高度化している。つまり、現在十分な強度をもっている暗号アルゴリズムが、今後もそうであるとは限らないのである。そこで、新しい解読技術に耐えられる新しいブロック暗号アルゴリズムを効率的に開発できるようにする必要がある。

【0006】そこで、従来において、ブロック暗号の開発支援手段の一例として、特開平5-14340号公報に見られるように、ブロック暗号の中間一致閉性試験（強度試験）を少ない記憶領域で実行可能とするものや、特開平8-190344号公報に見られるように、DESよりも探索計算量が多いFEAL等のinvolution型暗号へ実用的な時間で適用可能な暗号アルゴリズムの強度評価装置が知られている。

【0007】

【発明が解決しようとする課題】しかしながら、上記の強度試験あるいは強度評価は、暗号アルゴリズムの作成後に実施するものであり、基本的には、新しい解読技術に耐えられる新しいブロック暗号アルゴリズムを如何に効率良く開発するのを支援できるかが問題になる。

【0008】そのためには、ブロック暗号アルゴリズムの開発を効率化する手段が必要である。具体的には、

(1) ブロック暗号アルゴリズムを記述するための記法、(2) 該記法によるブロック暗号アルゴリズムの記述と編集を支援する手段、(3) 該記法によって記述されたブロック暗号アルゴリズムを試験的に実行する手段、(4) ブロック暗号アルゴリズムの強度を評価する手段等が必要である。

【0009】現在のところ、ブロック暗号アルゴリズムを記述するための記法としては、非形式的なデータフローダイアグラムが通常用いられているが、ブロック暗号アルゴリズムを細部まで正確に記述可能なものとはなっていない。これが第1の問題点である。

【0010】一方、暗号アルゴリズムは正確に実現されなければ、期待した効果を得ることができない。通常のブロック暗号アルゴリズムは、固定長のビット列に対して比較的単純な演算を繰り返し行なうだけであるので、アルゴリズムをそのまま反映したプログラムを作成するのは困難ではない。しかし、実際には高速化などの要求から何等かの最適化がなされることが多い。その場合、

4

プログラムが元のアルゴリズムを正確に実現していることを十分に確認する必要がある。したがって、ブロック暗号のプログラムをテストするために多くのテストケースが必要となる。

【0011】テストケースを作成するには、暗号アルゴリズムを実際に動作させて、平文とそれに対応する暗号文の組を作成する必要がある。しかし、従来は手計算でテストケースを作成しているため、多数の信頼できるテストケースを効率良く生成することが不可能であった。これが第2の問題点である。

【0012】一方また、前記の暗号アルゴリズム記法は、どちらかと言えばハードウェアによる実現に適した記法であり、ブロック暗号アルゴリズムを実現するプログラムの設計仕様としては不適当である。これが第3の問題点である。

【0013】さらに、前記のように、実用的なブロック暗号プログラムを作成するためには、最適化を行なう必要があるために、工数が掛かる。また、該ブロック暗号プログラムを組み込む応用プログラムを開発する場合、該ブロック暗号プログラムの開発が終了するまでは、スタブ（テスト用のダミーのサブルーチン）を使用することになる。したがって、その間は、応用プログラムをテストする際に、該ブロック暗号を使用することができない。これが第4の問題点である。

【0014】本発明の第1の目的は、予め定義したブロック暗号アルゴリズム記法に基づいてブロック暗号アルゴリズムの記述と編集を効率良く支援することができる暗号アルゴリズム開発支援装置を提供することにある。

【0015】本発明の第2の目的は、前記ブロック暗号アルゴリズム記法によって記述されたブロック暗号アルゴリズムを直接解釈実行することによって、該ブロック暗号アルゴリズムを実現するプログラムをテストする際に使用できるテストケースを自動的に生成し、暗号プログラムのテストを効率良く支援することができる暗号プログラム開発支援装置を提供することにある。

【0016】本発明の第3の目的は、前記ブロック暗号アルゴリズム記法によって記述されたブロック暗号アルゴリズムから、プログラム開発の目的に適したプログラムの設計仕様を自動生成し、暗号プログラムの生成を効率良く支援することができる暗号プログラム開発支援装置を提供することにある。

【0017】本発明の第4の目的は、前記ブロック暗号アルゴリズムから自動生成された前記プログラム設計仕様から、該ブロック暗号アルゴリズムを実現するプログラムを自動生成し、暗号プログラムの生成を効率良く支援することができる暗号プログラム開発支援装置を提供することにある。

【0018】本発明の第5の目的は、暗号プログラム開発のテスト工程において、ユーザによって指定されたスケジュールに従って、自動生成されたテストケースを用

いて、暗号プログラムを効率良くテストすることができ
る暗号プログラム開発支援装置を提供することにある。

【0019】

【課題を解決するための手段】上記の第1の目的を達成
するため、本発明は、予め定義したブロック暗号アルゴ
リズム仕様記述記法によるブロック暗号アルゴリズムの
ダイアグラム表現をユーザが記述及び編集するのを支援
する暗号ダイアグラム編集手段と、この暗号ダイアグラム
編集手段によって編集されたブロック暗号アルゴリズム
のダイアグラム表現を格納する暗号ダイアグラム格納
手段と、格納されたブロック暗号アルゴリズムのダイア
グラム表現を解釈実行する暗号ダイアグラム解釈実行手
段と、解釈実行中の該ブロック暗号アルゴリズムのダイ
アグラム表現に現れる変数の状態を格納する暗号ダイア
グラム実行環境格納手段と、このブロック暗号ダイアグ
ラム実行環境格納手段の内容を検査するための暗号ダイ
アグラム実行環境検査手段とを備えることを特徴とす
る。

【0020】ここで、本発明の前提となるブロック暗号
アルゴリズム仕様記述記法は、ビット列データを値とす
る変数のダイアグラム表現と、該ビット列データ上の演
算のダイアグラム表現とを構成要素として有し、前記変
数と前記演算を組み合わせるためのダイアグラム表現と
して、前記変数からの参照操作のダイアグラム表現と、
前記変数への代入操作のダイアグラム表現と、前記ビッ
ト列データへの前記演算の適用操作のダイアグラム表現
と、前記ビット列データの分割操作のダイアグラム表現
と、前記ビット列データの連結操作のダイアグラム表現
と、前記ビット列データの複製操作のダイアグラム表現
と、前記ビット列データへの繰り返し処理のダイアグラ
ム表現と、前記ビット列データ上の演算を定義するた
めのダイアグラム表現を備えることを特徴とする。

【0021】第2の目的を達成するために、本発明は、
テストケースを区別するためのIDと暗号化鍵と平文と
該平文を前記暗号化鍵によって暗号化した暗号文から
なるブロック暗号プログラムテストケースを格納するた
めの暗号プログラムテストケース格納手段と、前記IDを
生成するためのID計数手段と、前記暗号化鍵を生成す
るための鍵生成手段と、前記平文を生成するための平文
生成手段とを備え、前記暗号文を生成するブロック暗号
アルゴリズム開発支援手段とを備えることを特徴とす
る。

【0022】第3の目的を達成するために、本発明は、
ブロック暗号アルゴリズムのダイアグラム表現の構文を
解析するダイアグラム構文解析手段と、その解析結果で
あるダイアグラム構文木を格納するダイアグラム構文木
格納手段と、格納されたダイアグラム構文木からプログ
ラム設計仕様を生成する設計仕様生成手段と、ダイア
グラムとプログラム設計仕様との対応関係を定義した対
応テーブルと、生成された設計仕様を格納するための暗

号プログラム設計仕様格納手段とを備えることを特徴と
する。

【0023】第4の目的を達成するために、本発明は、
ブロック暗号プログラムの設計仕様の構文を解析する設
計仕様構文解析手段と、その解析結果である設計仕様構
文木を格納する設計仕様構文木格納手段と、格納された
設計仕様構文木からプログラムを生成するプログラム生
成手段と、設計仕様とプログラムの間の対応関係を定義
した対応テーブルと、生成されたプログラムを格納する
ための暗号プログラム格納手段とを備えることを特徴と
する。

【0024】第5の目的を達成するために、本発明は、
ブロック暗号プログラムのテストスケジュールを格納す
るテストスケジュール格納手段と、このテストスケジ
ュール格納手段に格納されたテストスケジュールに従っ
てテストケースを選択するテストケース選択手段と、ブ
ロック暗号プログラムを格納する暗号プログラム格納手
段と、格納されたブロック暗号プログラムに前記選択さ
れたテストケースの構成要素である鍵と平文を与えて実
行させるプログラム実行手段と、その実行結果である暗
号文と前記選択されたテストケースの構成要素である暗
号文とを比較するための比較手段と、前記選択されたテ
ストケースのIDと比較結果とからなるテスト結果を格納
するための暗号プログラムテスト結果格納手段とを備え
ることを特徴とする。

【0025】

【発明の実施の形態】以下、本発明の実施形態を図面
により詳細に説明する。

【0026】まず、実施形態の説明に先立ち、本発明の
各実施形態で使用するブロック暗号アルゴリズム仕様記
述記法について説明する。

【0027】[ブロック暗号アルゴリズム仕様記述記
法]図1および図2は、本発明の各実施形態で使用する
ブロック暗号アルゴリズム仕様記述法の文法の一覧を示
す図であり、ここで例示するブロック暗号アルゴリズム
仕様記述法は、図1(a)～(c)に示すようにビット
列データを値とする変数のダイアグラム表現11～13
と、図1(d)～(h)に示すようにビット列データ上
の演算のダイアグラム表現14～18とを構成要素とし
て有している。

【0028】さらに、前記変数と前記演算を組み合わ
せるためのダイアグラム表現として、図1(i)に示すよ
うに前記変数からの参照操作のダイアグラム表現19
と、図1(j)に示すように前記変数への代入操作のダ
イアグラム表現20と、図1(k)に示すように前記ビ
ット列データへの前記演算の適用操作のダイアグラム表
現21と、図1(m)に示すように前記ビット列データ
の分割操作のダイアグラム表現22と、図1(n)に示
すように前記ビット列データの連結操作のダイアグラム
表現23と、図1(p)に示すように前記ビット列デー

タの複製操作のダイアグラム表現24と、図2(a)に示すように前記ビット列データへの繰り返し処理のダイアグラム表現25と、図2(b)に示すように前記ビット列データ上の演算を定義するためのダイアグラム表現26を有する。

【0029】本例のブロック暗号アルゴリズム仕様記述法ではビット列データのみを扱う。変数は全てビット列データを格納するためにのみ使用され、演算はビット列データに対するもののみが使用可能である。

【0030】例えば、図1(a)のダイアグラム表現11は「64ビットデータ」を格納するための「B」という名前の変数を表している。図1(b)のダイアグラム表現12は「32ビットデータ」を格納するための「L_0」という名前の添字付変数を表しており、変数名中の「0」が添字である。図1(c)のダイアグラム表現13は「32ビットデータ」を格納するための16個($i=1\sim 16$)の添字付変数「L_1」～「L_16」を表している。図1(d)のダイアグラム表現14は「f」という名前の演算を表している。

【0031】図1(e)～(h)は、よく使われる演算の例であり、このような特殊な演算は必要に応じて導入される。図1(e)のダイアグラム表現15は、2つのビット列データの排他的論理和を表している。図1

(f)のダイアグラム表現16は、「64ビットデータ」を「56ビットデータ」に縮小転写する演算を表している。図1(g)のダイアグラム表現17は、ビット列データを「KSビット」だけシフトする演算を表している。図1(h)のダイアグラム表現18は、「2ビットデータ」と「4ビットデータ」に対して「4ビットデータ」を代入する演算を表している。

【0032】図1(i)のダイアグラム表現19は、変数中に格納されているビット列データを参照するための表現であり、特に記述の対象となっているアルゴリズムの入出力データを格納する変数には「input」または「output」というラベルを付ける。

【0033】図1(j)のダイアグラム表現20は、変数「CB」に「64ビット列データ」を代入するための表現である。図1(k)のダイアグラム表現21は、ビット列データに対して「f」という名前の演算を適用するための表現である。

【0034】図1(m)のダイアグラム表現22は、ビット列データを分割し、複数ビット列データとして変数X、Yに格納するための表現である。図1(n)のダイアグラム表現23は、複数個のビット列データを連結し、1個のビット列データとするための表現である。図1(p)のダイアグラム表現24は、ビット列データを複製し、複数の同じビット列データとするための表現である。

【0035】図2(a)のダイアグラム表現25は、 i 個($i=1\sim 16$)の添字付きの変数Xに格納されてい

るビット列データに対して、同じ処理を繰り返し適用し、その結果を別の添字付きの i 個($i=1\sim 16$)の変数Yに格納するための表現である。点線内は繰り返しの内容であり、前記の表現と、場合によってはさらに繰り返しの表現を用いて記述される。繰り返し回数は左上に「16 times」というように記述する。

【0036】点線内の左側にある「 i 」は、現在何度目の繰り返しであるかを表す繰り返し変数である。この繰り返し変数「 i 」は「1」から始まって、順に「1」ずつ加算され、繰り返し回数に達したところで繰り返し処理は全て終了する。

【0037】図2(b)のダイアグラム表現26は、演算を定義するための表現である。この表現によって定義された演算は、ダイアグラム表現14の表現によって他の場所で使用することができる。点線内は演算の内容であり、前記の表現を用いて記述される。

【0038】同名の変数の異なる出現は同一の変数を表す。変数へのデータの格納は1回のみ許される。変数の参照は、何回行なっても構わないが、データの格納されていない変数の参照は許されない。

【0039】〔記述例〕図3～図5は、ブロック暗号アルゴリズムである「DES」を本発明のブロック暗号アルゴリズム仕様記述法によって記述したアルゴリズム構成図であり、図3はブロック暗号アルゴリズムDESの全体を表し、図4は図3で使われている演算「f」112の処理の詳細を表し、図5は図3で使われている変数「K_ i 」($i=1\sim 16$)109に格納される16個の鍵Kの生成処理を詳細に表したものである。

【0040】これらの図3～図5で使用されている転写、シフト、代入の演算の詳細は、例えば図6に示すように別途与えられる。図6(a)は図3で使われている転写演算「IP」の詳細であり、例えば、出力ビット列の第1ビットが入力ビット列の第58ビットである。図6(b)は図4で使われている拡大転写演算「EP」の詳細であり、例えば、出力ビット列の第1ビットが入力ビット列の第32ビットである。図6(c)は図4で使われている代入演算「S_1」の詳細であり、2ビット入力で表される4通りの場合と4ビット入力の16通りの場合の組合せをマトリクスで表現しており、出力の4ビット列の値は該マトリクスの各枠目の値となる。図6(d)は図5で使われているシフト演算の詳細であり、例えば、「K_1」は入力ビット列を1ビット左シフトすることを表している。

【0041】図3のアルゴリズムの詳細は次のようになる。

【0042】まず、暗号化の対象となる平文データは、「B」という名前の64ビット変数101に格納される。この変数「B」には「input」というラベルが付けられており、該アルゴリズムに対する入力であることが示されている。平文データは変数「B」の参照を表

す矢印102を経て、「IP」という名前の64ビットデータに対する転字103に送られる。

【0043】この転字処理結果である64ビットデータは、データの分割を表す矢印104を経て、上位の32ビットデータは「L_0」という名前の32ビット変数105に、下位の32ビットデータは「R_0」という名前の32ビット変数107に格納される。

【0044】次に、繰り返し処理110に入り、この処理内部において同様の処理が16回（「16 times」）行なわれる。

【0045】第1（ $i=1 \dots 16$ ）回目の処理では、「L_（ $i-1$ ）」という名前の32ビット変数105または106と、「R_（ $i-1$ ）」という名前の32ビット変数107または108と、「K_i」という名前の48ビット変数109のデータが参照され、処理結果は「L_i」という名前の32ビット変数114または115と、「R_i」という名前の32ビット変数116または117に格納される。

【0046】この繰り返し処理110の内容の詳細は次のようなる。変数「R_（ $i-1$ ）」はデータの複製を表す矢印111を経て、一方は変数「L_i」114または115に格納され、他方は演算「f」112の二つの入力うちの1つになる。

【0047】演算「f」112のもう1つの入力変数は「K_i」である。この演算「f」112の結果は、入力変数「L_（ $i-1$ ）」と共に排他的論理和113の入力となる。この排他的論理和113の結果は出力変数「R_i」116または117に格納される。

【0048】このような繰り返し処理が全て終了すると、「L_16」という名前の32ビット変数115と「R_16」という名前の32ビット変数117のデータが、データの連結を表す矢印118を経て連結されて64ビットデータとなり、「IP⁻¹」（転字IPの逆）という転字119に入力される。

【0049】この転字119の処理結果が暗号文であり、データの格納を表す矢印120を経て、「CB」という名前の64ビット変数121に格納される。この変数121には「output」というラベルが付けられており、DESアルゴリズムからの出力であることが示されている。

【0050】なお、図3には、変数「L_i（ $i=1 \dots 15$ ）」の2つの出現106と115があるが、同一の変数を表している。変数「R_i（ $i=1 \dots 15$ ）」についても同様である。

【0051】図4の演算「f」の定義112の詳細は次の通りである。演算「f」では32ビットデータ「R」132と48ビットデータ「K」133の2つの入力から、32ビットデータ「f（R, K）」149を計算する。

【0052】入力データ「R」132は、32ビットデ

ータの48ビットデータへの拡大転字「EP」134によって、48ビットデータ135に変換され、もう1つの48ビットデータ「K」133と共に排他的論理和136の入力となる。この排他的論理和136の演算結果137は、8個（ $j=1 \dots 8$ ）の6ビットデータに分割され、8個の6ビットデータ変数「X_j」（ $j=1 \dots 8$ ）」138に格納される。データの格納を表す矢印137は、ここでは添字付変数の並びに接続されているため、データの分割を表している。

10 【0053】次に、繰り返し処理139に入り、8個の6ビットデータ変数「X_j」（ $j=1 \dots 8$ ）のそれぞれに対して次の処理が行なわれる。

【0054】まず、6ビットデータ変数「X_j」138は1ビット、4ビット、1ビットのデータに分割され、それぞれ、1ビットデータ変数「r1」140、4ビットデータ変数「c」141、1ビットデータ変数「r2」142に格納される。ここで、変数「r1」140、変数「c」141、変数「r2」142は、繰り返し変数「j」が加算される毎に新しく用意される。

20 【0055】変数「r1」140と「r2」141のデータを連結した2ビットデータ143と、変数「c」142の4ビットデータ144が代入演算「S_j」145に入力される。この演算結果は4ビット変数「Y_j」146に格納される。

【0056】このような繰り返し処理が全て終了すると、8個の4ビットデータ「Y_j」（ $j=1 \dots 8$ ）」146が連結されて32ビットデータ147となり、32ビットデータに対する転字「P」148に入力される。データ147の参照を表す矢印は、ここでは添字付変数の並びからの参照になっているため、データの連結を表している。この演算結果が「f（R, K）」149となる。

30 【0057】図5に示す変数「K_i（ $i=1 \dots 16$ ）」167の生成処理の詳細は次の通りである。まず、DESアルゴリズムに対して入力される鍵（K）が64ビットデータ変数「K」151に格納される。この64ビットデータ変数「K」151は、矢印152を経て、64ビットから56ビットへの縮小転字「KP」153に入力され、56ビットデータ154に変換される。

40 【0058】この56ビットデータ154は、2つの28ビットデータに分割され、それぞれ28ビット変数「KL_0」155と「KR_0」157に格納される。

【0059】次に、繰り返し処理（1）159に入り、各「KL_（ $i-1$ ）」（ $i=1 \dots 16$ ）」155あるいは156と、「KR_（ $i-1$ ）」（ $i=1 \dots 16$ ）」157あるいは158に対して、次の処理が行なわれ、「KL_i（ $i=1 \dots 16$ ）」162と「KR_i（ $i=1 \dots 16$ ）」163が順次生成される。

【0060】各「KL_ (i-1)」155あるいは156はシフト演算「KS_i」160の入力となる。この演算結果は変数「KL_i」162に格納される。

【0061】また、各変数「KR_ (i-1)」157あるいは158はシフト演算「KS_i」161の入力となる。この演算結果は変数「KR_i」163に格納される。

【0062】上記の繰り返し処理の後であるいは並行して、各「KL_i (i=1..16)」162と「KR_i (i=1..16)」163に対して、次の繰り返し処理(2)164が行なわれる。

【0063】各「KL_i」162と「KR_i」163を連結した56ビットデータ165に対して、56ビットから48ビットへの縮小転写「PC」166が適用され、その結果が48ビット変数「K_i」167に格納される。

【0064】この記述例が示すように、本例のブロック暗号アルゴリズム記述記法によれば、ブロック暗号アルゴリズムの詳細を厳密にかつ簡潔に記述することができる。

【0065】【第1の実施形態】次に、上述したブロック暗号アルゴリズム仕様記述記法を使用して実現される暗号アルゴリズム開発支援装置の実施形態について、図面に従って具体的に説明する。

【0066】図7は、ブロック暗号のアルゴリズム開発支援装置の実施形態を示す機能構成図である。この第1の実施形態のブロック暗号のアルゴリズム開発支援装置は、前述した暗号アルゴリズムのダイアグラム表現をユーザが記述及び編集するのを支援する暗号ダイアグラム編集装置201と、記述および編集されたブロック暗号アルゴリズムのダイアグラム表現を格納する暗号ダイアグラム格納装置204と、ブロック暗号アルゴリズムのダイアグラム表現を解釈実行する暗号ダイアグラム解釈実行装置202と、解釈実行中のブロック暗号アルゴリズムのダイアグラム表現に現れる変数の状態を格納する暗号ダイアグラム実行環境格納装置205と、暗号ダイアグラム実行環境格納装置205の内容を検査するための暗号ダイアグラム実行環境検査装置203とから構成される。

【0067】暗号ダイアグラム編集装置201は、図3で示したようなブロック暗号アルゴリズムのダイアグラム表現をユーザが記述及び編集するのを支援するためのものであり、従来のダイアグラム編集技術によって実現可能である。この暗号ダイアグラム編集装置201は、図6のような演算の詳細を定義するための機能も持っており、この定義は暗号ダイアグラム格納装置204に格納される。

【0068】暗号ダイアグラム解釈実行装置202は、ブロック暗号アルゴリズムのダイアグラム表現を解釈実行するためのものであり、暗号ダイアグラム格納装置2

04に格納されている暗号ダイアグラムの構成要素を計算可能なものから順に計算して、その計算結果である値を暗号ダイアグラム実行環境格納装置205に格納していく。ただし、ダイアグラムの構成要素が「計算可能である」ことは次のように定義される。

【0069】【構成要素の計算可能性の定義】

(a) アルゴリズムの入力変数は計算可能である。

【0070】(b) 組み込み演算はその入力が全て値が決まっていれば、出力は計算可能である。

10 【0071】(c) 変数は代入される値が決まっていれば、計算可能である。

【0072】(d) n回の繰り返し部の第1回目は、繰り返しの添字を「1」としたときに、入力変数全ての値が決まっていれば、計算可能である。

【0073】(e) n回の繰り返し部の第i (i=2..n) 回目は、第1回目から第i-1回目の出力変数が決まっていれば、繰り返しの添字をiとしたときに入力変数全ての値が決まっていれば、計算可能である。

20 【0074】(f) ユーザ定義演算は、入力変数全ての値が決まっていれば、計算可能である。

【0075】暗号ダイアグラム解釈実行装置202は、図8の詳細構成図に示すように、暗号ダイアグラム格納装置204に格納されている暗号ダイアグラムを読み取るためのダイアグラム走査装置206と、暗号ダイアグラムの走査結果に基づき当該暗号ダイアグラムの構成要素の計算可能性を検査するための計算可能性検査装置207と、暗号アルゴリズムの入力変数を設定するための初期値設定装置208と、暗号ダイアグラムの構成要素の値を計算するための値計算装置209から構成される。

【0076】暗号ダイアグラム実行環境検査装置203は、ユーザが暗号ダイアグラム実行環境格納装置205の内容を検査するためのものであり、図9の詳細構成図に示すように、暗号ダイアグラムの構成要素のうち、ユーザから指定された要素の値を読み取るための検査対象指定装置210と、読み取った値を表示するための検査結果表示装置211から成る。

【0077】この第1の実施形態の暗号アルゴリズム開発支援装置の基本的な使用方法は、次のようになる。

40 【0078】まず、ユーザは、暗号ダイアグラム編集装置201を使用し、図3に示したような記法によるブロック暗号アルゴリズムのダイアグラム表現を記述及び編集し、暗号ダイアグラム格納装置204に格納する。

【0079】次に、暗号ダイアグラム格納装置204に格納したブロック暗号アルゴリズムがユーザの意図した通りに動作することを確認するために、暗号ダイアグラム解釈実行装置202を使用し、実際に前記ブロック暗号アルゴリズムを実行させる。この実行結果は、暗号ダイアグラム実行環境格納装置205に、図11～図13 50 に示すような表として格納される。

【0080】最後に、ユーザは暗号ダイアグラム実行環境検査装置203を使用して前記実行結果を詳細に調べる。

【0081】以下、暗号ダイアグラム解釈実行装置202の動作を図10に示すフローチャートに基づき1ステップずつを詳細に説明する。

【0082】ステップ212

暗号ダイアグラム格納装置204に格納されているダイアグラムの構成要素を、ダイアグラム走査装置206によって調べ、暗号ダイアグラム実行環境格納装置205を初期化する。暗号ダイアグラム実行環境は、アルゴリズム全体に対する表の他、各繰り返し部の各回に対応した表や、ユーザ定義演算に対応する表が作られて初期化される。

【0083】例えば、図3～図5に示したDES暗号アルゴリズムの生成処理の場合、図11(a)に示す暗号アルゴリズムDESの生成処理全体の表1100、図11(b)に示すDES全体の繰り返し部139に関する表1110、図12に示す鍵生成処理全体の表1120、図13(a)に示す鍵生成処理中の第1の繰り返し部の各回の表1130、鍵生成処理中の第2の繰り返し部の各回の表1140が作られて初期化される。

【0084】ここで、これらの各表は、「構成要素」1101、「計算可能か?」1102、「値」1103の各欄から構成され、「構成要素」1101の欄にはブロック暗号アルゴリズム中に記述した変数等のダイアグラム表現の名称が設定される。また、「計算可能か?」1102の欄には初期段階では「NO」が設定され、計算可能になった段階で「YES」が設定されるようになっている。さらに、「値」1103の欄には、変数等のダイアグラム表現の計算結果が設定されるようになっている。

【0085】ステップ213

計算可能性検査装置207によって、前記の計算可能性の定義に従って、ダイアグラム表現の中で値1103が計算可能なものを全て列挙する。このとき、構成要素間の関係はダイアグラム走査装置206によって調べる。

【0086】ステップ214

ステップ213で列挙された計算可能な構成要素の値1103を値計算装置209によって計算する。このとき、ダイアグラム走査装置206によって、計算に必要な値を持つ他の構成要素を調べ、その値は暗号ダイアグラム実行環境格納装置205から参照する。また、その構成要素がアルゴリズムへの入力変数である場合は、初期値設定装置208によって、ユーザによって指定された値を計算値とする。

【0087】ステップ215

暗号アルゴリズムの全ての出力変数に値が格納されたかを調べる。そうであれば、実行を終了する。さもなければ、ステップ213に戻る。

【0088】例えば、図5に示す暗号アルゴリズムDESの鍵生成処理において*i*=1の鍵である出力変数「K_1」167が生成されるまでの表1120、1130、1140の内容は図14～図16に示すようなものとなる。

【0089】すなわち、実行の開始直後のステップ212で暗号ダイアグラム実行環境格納装置205が初期化された時点では、条件が揃っていないために構成要素は全て計算可能ではない。図14～図16では、初期化された時点で計算可能か否かということを「開始」の欄1401に「NO」で示している。

【0090】次に、ステップ213で、計算可能性検査装置207によって前記の計算可能性の定義に従って、アルゴリズムの入力変数Kのみが計算可能な構成要素として列挙され、ステップ214で、値計算装置209と初期値設定装置208によって、ユーザによって指定された値XがKの計算値となる。図14では、計算段階の欄1402の(1)に入力変数K=Xになったことを示している。また、この時点で入力変数Kの「計算可能か?」が「YES」に更新されたことを示している。

【0091】ステップ215で、暗号アルゴリズムの全ての出力変数に値が格納されたかを調べるが、そうではないのでステップ213に戻る。2度目のステップ213では、入力変数Kの値が決まっているので、計算可能性検査装置207によって、前記の計算可能性の定義に従って転字演算KPが計算可能な構成要素として列挙される。そして、転字演算KPの「計算可能か?」が「YES」に更新される。

【0092】ステップ214で、値計算装置209によってKP(X)が計算され、KPの値となる。図14では、計算段階の欄1402の(2)にKPの値が求められたことをKP(X)で示している。続く、3度目のステップ213とステップ214では「KL_0」と「KR_0」の値が計算される。図14では、計算段階の欄1402の(3)に「KL_0」と「KR_0」の値が求められたことを示している。

【0093】4度目のステップ213とステップ214では、繰り返し部(1)の*i*=1の場合が計算可能になり、その入力変数「KL_0」と「KR_0」の値が決まる。図15では、計算段階の欄1402の(4)に「KL_0」と「KR_0」の値が求められたことを示している。

【0094】5度目のステップ213とステップ214では、2つの「KS_1」の値が計算され、6度目のステップ213とステップ214では、繰り返し部(1)の*i*=1の場合の出力変数である「KL_1」と「KR_1」の値が決まる。図15では、計算段階の欄1402の(5)に「KS_1」の値が求められたことを示し、また計算段階の欄1402の(6)に「KL_1」と「KR_1」の値が求められたことを示している。

【0095】以下同様にして、図5のダイアグラム表現で記述された暗号アルゴリズムの実行が続けられる。これによって、12度目のステップ213とステップ214で第1番目($i=1$)の出力変数「K_1」の値が決まる。図16では、計算段階の欄1402の(12)に「K_1」の値が求められたことを示している。

【0096】このような処理が $i=16$ になるまで繰り返して実行される。これによって、16個の鍵に相当する変数「K_1」が生成される。なお、暗号アルゴリズムDES全体の処理としては、変数「CB」の値が求めた段階で終了する。変数「CB」の値が求めたならば、図11の表1100の「計算可能か?」1102の欄は、全て「YES」になる。

【0097】このように、第1の実施形態によれば、前述した記法によるブロック暗号アルゴリズムのダイアグラム表現をユーザが記述及び編集するのを支援できる上に、そのブロック暗号アルゴリズムのダイアグラム表現を解釈実行し、当該ブロック暗号ダイアグラム実行環境を詳細に調べることができ、ブロック暗号アルゴリズム開発を効率的に行なうことができる。

【0098】〔第2の実施形態〕次に、本発明を適用した暗号プログラムテストケース生成装置の実施形態について、図面に従って具体的に説明する。

【0099】図17は、暗号プログラムテストケース生成装置の実施形態を示す機能構成図である。この実施形態の暗号プログラムテストケース生成装置300は、テストケースを区別するためのID(識別情報)3041と、暗号化鍵3042および平文3043並びに該平文3043を暗号化鍵3042によって暗号化した暗号文3044とからなるブロック暗号プログラムテストケース304を格納するための暗号プログラムテストケース格納装置305と、前記ID3041を生成するためのID計数装置301と、前記暗号化鍵3042を生成するための鍵生成装置302と、前記平文3043を生成するための平文生成装置303とから構成され、前記暗号文3044を生成するために、前述の第1の実施形態の暗号アルゴリズム開発支援装置200を利用する。

【0100】この実施形態の暗号プログラムテストケース生成装置300の基本的な使用方法是次のようになる。

【0101】ユーザは、前述した暗号アルゴリズム開発支援装置200を利用し、予めブロック暗号アルゴリズムを開発しておく。すると、ユーザは、該ブロック暗号アルゴリズムを実現するプログラムを検査する際に使用するテストケースを、本実施形態の暗号プログラムテストケース生成装置300を使用して作成することができる。

【0102】まず、ID計数装置301によって新しいテストケースのID3041を生成する。次に、鍵生成装置302によって暗号化鍵3042を生成し、さらに

平文生成装置303によって暗号化の対象となる平文3043を生成する。

【0103】鍵生成装置302と平文生成装置303は乱数等の手段によって、暗号化鍵3042や平文3043を生成する。この場合、ユーザが暗号化鍵3042や平文3043を直接入力することもある。

【0104】暗号化鍵3042と平文3043を生成したならば、暗号アルゴリズム開発支援装置200のブロック暗号ダイアグラム解釈実行装置203に暗号化鍵3042と平文3043とを入力し、前記第1の実施形態で説明した手順によって、前記暗号アルゴリズムを解釈実行することにより、前記平文3043を前記暗号アルゴリズムにより暗号化した暗号文3044が得られる。

【0105】そこで、前記ID3041と暗号化鍵3042および平文3043と暗号文3044を一組にし、テストケース304として暗号プログラムテストケース格納装置305に格納蓄積する。

【0106】このように、第2の実施例によれば、新しく開発されたブロック暗号アルゴリズムを実現するプログラムをテストするためのテストケースを効率良く機械的に生成することができる。

【0107】〔第3の実施形態〕次に、本発明の第3の実施形態であるブロック暗号プログラム設計仕様生成装置について、図面に従って具体的に説明する。

【0108】図18は、ブロック暗号プログラム設計仕様生成装置の機能構成図であり、この実施形態のブロック暗号プログラム設計仕様生成装置400は、前記の暗号アルゴリズム開発支援装置200で作成したブロック暗号アルゴリズムのダイアグラム表現の構文を解析するダイアグラム構文解析装置401と、その解析結果であるダイアグラム構文木を格納するダイアグラム構文木格納装置403と、この格納装置403に格納されたダイアグラム構文木からプログラム設計仕様を生成する設計仕様生成装置402と、ダイアグラムとプログラム設計仕様の間の対応関係を定義したダイアグラム設計仕様対応テーブル405と、生成された設計仕様を格納するための暗号プログラム設計仕様格納装置404とから構成される。

【0109】この第3の実施形態のブロック暗号プログラム設計仕様生成装置の動作の詳細は次のようになる。

【0110】まず、前記ブロック暗号アルゴリズム開発支援装置200の暗号ダイアグラム格納装置204に格納されている暗号ダイアグラムを、ダイアグラム構文解析装置401によって構文解析する。

【0111】その解析結果は、ダイアグラム構文木格納装置403に格納される。続いて、ダイアグラム設計仕様対応テーブル405に従って、設計仕様生成装置402によって、ダイアグラム構文木からプログラム設計仕様を生成する。その生成された設計仕様は暗号プログラム設計仕様格納装置404に格納される。

【0112】プログラム設計仕様の記述記法としてPAD (Problem Analysis Diagram) を使う場合、ダイアグラム設計仕様対応テーブル405の詳細は例えば図19および図20のように定義されている。

【0113】図19および図20において、項番(1)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される変数があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるような変数の宣言が生成されることを表している。

【0114】項番(2)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される添字付き変数の並びがあった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるような添字付き変数の並びの宣言が生成されることを表している。

【0115】項番(3)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される転字演算があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるような転字演算の定義が生成されることを表している。この定義内容は暗号ダイアグラム格納装置204内にある図6(a)、(b)のような転字演算の詳細である。

【0116】項番(4)は、ダイアグラム記述中にダイアグラム表現の欄4052に示されるシフト演算があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるようなシフト演算の定義が生成されることを表している。この定義内容は暗号ダイアグラム格納装置204内にある図6(c)のようなシフト演算の詳細である。

【0117】項番(5)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される代入演算があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるような代入演算の定義が生成されることを表している。この定義内容は暗号ダイアグラム格納装置204内にある図6(c)のような代入演算の詳細である。

【0118】項番(6)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される変数の参照を表す表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるような実行文中の式の部分的な式として、該変数の参照が生成されることを表している。

【0119】項番(7)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される変数への代入を表す表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるように、それまでの計算結果を該変数へ代入する実行文が生成されることを表している。

【0120】項番(8)は、ダイアグラム記述中にダイ

アグラム表現の欄4052に示される演算の適用を表す表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるように、実行文中の式の部分的な式として、それまでの計算結果に該演算を適用する式が生成されることを表している。

【0121】項番(9)は、ダイアグラム記述中にダイアグラム表現の欄4052に示されるデータの分割を表す表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるように、それまでの計算結果を複数の変数へ代入する実行文が生成されることを表している。

【0122】項番(10)は、ダイアグラム記述中にダイアグラム表現の欄4052に示されるデータの連結を表す表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるように、実行文中の式の部分的な式として、それまでの計算結果を表す複数の式を連結する表現が生成されることを表している。

【0123】項番(11)は、ダイアグラム記述中にダイアグラム表現の欄4052に示されるデータの複製を表す表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるように、実行文中の式の部分的な式として、それまでの計算結果を表す式が複数個生成されることを表している。

【0124】項番(12)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される繰り返しを表す表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるように、実行文中の式の部分的な式として、PADの繰り返し構造の実行文が生成されることを表している。

【0125】項番(13)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される入力変数と出力変数の表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるように、該入力変数を引数とし、該出力変数の値を返値とするPADの演算定義の表現が生成されることを表している。

【0126】項番(14)は、ダイアグラム記述中にダイアグラム表現の欄4052に示される演算定義の表現があった場合、生成されるプログラム設計仕様では、設計仕様(PAD)表現の欄4053に示されるように、PADの演算定義の表現が生成されることを表している。

【0127】この図19および図20で例示した対応テーブル405は再帰的に適用される。例えば図3、図4、図5のDESアルゴリズムの設計仕様を生成したならば、この対応テーブル405に従って、図21および図22に示すようなプログラム設計仕様が得られる。

【0128】ここで、図21のPAD表現によるプログラム仕様は、図3と図5のダイアグラム記述を合わせたものに対応している。例えば、図21の演算「des_main」の定義411、出力変数CBの宣言412、出力変数CBを演算結果として返す表現426は、図3の「入力変数B」101、図5の「入力変数K」151と図3の「出力変数CB」121に図20の項番(15)の定義を適用した結果である。

【0129】同様に、図21の変数及び種々の演算の宣言413は、図3の「転字IP」103、変数の並び「L_i (i=0..16)」105と106と115、変数の並び「R_i (i=0..16)」107と108と117、変数の並び「K_i (i=1..16)」109、図5の「転字KP」153、変数の並び「KL_i (i=0..16)」155と162、変数の並び「KL_i (i=0..16)」157と163、シフト演算の並び「KS_i (i=1..16)」160、「転字PC」166に、図19の項番(1)、(2)、(3)、(4)を適用した結果である。

【0130】図21の414~419は、図5の鍵生成部から生成されたステップである。図21の鍵生成の第1ステップ414は、図5のデータの「参照」152に図19の項番(6)を適用し、データの「分割」154に図19の項番(9)を適用して生成されたステップである。

【0131】図21の鍵生成の第2ステップ(繰返し(1))415は、図5の「繰返し処理(1)」159に図20の項番(12)を適用して生成されたステップである。また、繰返し(1)の第1サブステップ416は、変数「KL_i (i-1)」の参照に図19の項番(6)を、シフト演算「KS_i」160の適用に図19の項番(8)を、変数「KL_i」への代入に図19の項番(7)を適用して生成されたステップである。繰返し(1)の第2サブステップ417も同様である。鍵生成の第3ステップ(繰返し(2))418と、そのサブステップ419についても同様である。

【0132】図21の420~425は、図3のDESの全体の記述から生成されたステップであり、その生成過程は前記の鍵生成部の場合と同様である。

【0133】図22の演算「f」の定義427と演算「f」の計算値を返すステップ433は、図4の演算「f」の定義139に図20の項番(14)を適用して生成されたものである。ただし、演算「f」の計算値を返すステップ433では、図4のデータの連結147に図19の項番(10)を、「転字演算P」148の適用に図19の項番(8)を適用して得られた表現が演算「f」の計算値となっている。

【0134】図22の他のステップ428~432は、前記の例と同様にして生成される。

【0135】このように、ユーザは、本実施形態の暗号

プログラム設計仕様生成装置400を使用し、ブロック暗号アルゴリズム開発支援装置200を利用して開発されたブロック暗号アルゴリズムを実現するプログラムの設計仕様を自動的に生成することができる。

【0136】なお、本実施形態では、プログラム設計仕様表現としてPADを用いたが、フローチャート等の他のプログラム設計仕様表現を使用した場合でも、同様にして容易に実現可能である。

【0137】[第4の実施形態]次に、本発明の第4の実施形態であるブロック暗号プログラム生成装置について、図面に従って具体的に説明する。

【0138】図23は、暗号プログラム生成装置500の機能構成図であり、この実施形態の暗号プログラム生成装置500は、暗号プログラム設計仕様生成装置400の暗号プログラム設計仕様格納装置404に格納されたブロック暗号プログラム設計仕様の構文を解析する設計仕様構文解析装置501と、その解析結果である設計仕様構文木を格納する設計仕様構文木格納装置503と、この格納装置503に格納された設計仕様構文木からプログラムを生成するプログラム生成装置502と、設計仕様とプログラムの間の対応会計を定義した設計仕様プログラム対応テーブル505と、生成されたプログラムを格納するための暗号プログラム格納装置504から構成される。

【0139】この第4の実施形態のブロック暗号プログラム生成装置の動作の詳細は次のようになる。

【0140】まず、前記暗号プログラム設計仕様生成装置400の暗号プログラム設計仕様格納装置404に格納されている暗号プログラム設計仕様を、設計仕様構文解析装置501によって、構文解析する。

【0141】その解析結果は、設計仕様構文木格納装置503に格納される。続いて、設計仕様プログラム対応テーブル505に従って、プログラム生成装置502によって、前記設計仕様構文木からプログラムを生成する。生成されたプログラムは暗号プログラム格納装置504に格納される。

【0142】生成された暗号プログラムをコンパイラ/リンカ506によってコンパイルし、ライブラリ507とリンクすることにより、実行可能形式暗号プログラム508が得られる。ライブラリ507には、生成された暗号プログラムで使用されるビットデータの型や手続きが含まれる。

【0143】プログラム設計仕様の記述記法としてPAD (Problem Analysis Diagram) を使い、プログラミング言語としてC言語を使う場合、設計仕様プログラム対応テーブル505の詳細は例えば図24および図25のように定義されている。

【0144】項番(1)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される変数宣言があった場合、生成されるプログラムでは、C言語表現

の欄5053に示すように、対応するビット数の型の変数宣言が生成されることを表している。

【0145】項番(2)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される変数並びの宣言があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、対応するビット数の型の配列変数の宣言が生成されることを表している。

【0146】項番(3)は、プログラム設計仕様中に転字定義があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、int型配列変数の定義が生成されることを表している。

【0147】項番(4)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示されるシフト定義があった場合、生成されるプログラムでは、int型変数の定義が生成されることを表している。

【0148】項番(5)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される代入定義があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、int型2次元配列変数の定義が生成されることを表している。ただし、例えば入力力が2ビットと4ビットであれば、該配列の大きさは4(=2の2乗)×16(=2の4乗)となる。

【0149】項番(6)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される変数の参照を表す表現があった場合、生成されるプログラムでは、文中の式の部分的な式として、該変数の参照が生成されることを表している。

【0150】項番(7)は、項番(6)と同様であるが添字付変数に対応するのは、配列要素の参照であることを表している。

【0151】項番(8)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される変数への代入を表す表現があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、代入文が生成されることを表している。

【0152】項番(9)は、項番(8)と同様であるが、添字付変数に対応するのは、配列要素への代入であることを表している。

【0153】項番(10)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される演算の適用を表す表現があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、実行文中の式の部分的な式として、該演算を適用する式が生成されることを表している。

【0154】項番(11)～(15)は、項番(10)と同様であるが、排他的論理和、転字、逆転字、シフト、代入の各演算では、該演算に対応するC関数を用いた特殊な式が対応することを表している。

【0155】項番(16)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示されるデータ

の分割を表す表現があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、「divide」というC関数を用いたデータ分割を表す実行文が生成されることを表している。

【0156】項番(17)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示されるデータの連結を表す表現があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、実行文中の式の部分的な式として、「concat」というC関数を用いたデータ連結を表す式が生成されることを表している。

【0157】項番(18)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される繰り返しを表す表現があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、「for文」が生成されることを表している。

【0158】項番(19)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される値を返す(返値)表現があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、「return文」が生成されることを表している。

【0159】項番(20)は、プログラム設計仕様中に設計仕様(PAD)表現の欄5052に示される演算定義の表現があった場合、生成されるプログラムでは、C言語表現の欄5053に示すように、関数の宣言と定義が生成されることを表している。この図24および図25に示す内容の対応テーブル405は再帰的に適用される。

【0160】プログラミング言語としてC言語を使う場合、前記ライブラリ507のヘッダーファイルは例えば図27に示すようになる。この中では、図24および図25に示した内容の対応テーブル405のC言語表現で使用されているデータ型や関数のプロトタイプが宣言されている。

【0161】図24および図25に示した内容の対応テーブル405に従って、例えば、図21のDESアルゴリズムのプログラムを生成すると、図26(a)、(b)に示すようなプログラム4041、4042が得られる。

【0162】図26(a)は、図21で示した設計仕様から生成されたプログラムであり、同図(b)は図22で示した設計仕様から生成されたプログラムである。

【0163】このように、本実施形態においては、前述したブロック暗号アルゴリズム開発支援装置200とブロック暗号プログラム設計仕様生成装置400を利用して開発されたブロック暗号アルゴリズムを実現するプログラムを自動的に生成することができる。この生成されたブロック暗号プログラムは、該ブロック暗号を使用する応用プログラムの開発時にスタブとして使用することができる。

【0164】なお、本実施形態では、プログラム設計仕様表現としてPADを用い、プログラミング言語としてC言語を用いたが、フローチャート等の他のプログラム設計仕様表現と、C++やPascal等の他のプログラミング言語を使用した場合でも、同様にして実現可能である。

【0165】〔第5の実施形態〕次に、本発明の第5の実施形態であるブロック暗号プログラムテスト支援装置について、図面に従って具体的に説明する。

【0166】図28は、ブロック暗号プログラムテスト支援装置の機能構成図であり、この実施形態のブロック暗号プログラムテスト支援装置600は、ブロック暗号プログラムのテストスケジュールを格納するテストスケジュール格納装置601と、格納されたテストスケジュールに従ってテストケース304を選択するテストケース選択装置602と、ブロック暗号プログラムを格納する暗号プログラム格納装置605と、格納されたブロック暗号プログラムに対し前記選択されたテストケース304の構成要素である暗号鍵3042と平文3043を与えて実行させるプログラム実行装置603と、その実行結果である暗号文と前記選択されたテストケース304の構成要素である暗号文3044とを比較するための比較装置604と、前記選択されたテストケース304の構成要素であるID3041と比較装置604の比較結果とからなるテスト結果607を格納するための暗号プログラムテスト結果格納装置606とから構成される。

【0167】この第5の実施形態のブロック暗号プログラムテスト支援装置600の動作の詳細は次のようになる。

【0168】まず、ユーザはあらかじめ、前述したブロック暗号プログラムテストケース生成装置300によって、暗号プログラムのテストケース304を生成しておく。

【0169】また、テスト対象となる暗号プログラムを暗号プログラム格納装置605に格納しておき、図29に示すようなテストスケジュール6010をテストスケジュール格納装置601に格納しておく。すると、本実施形態のブロック暗号プログラムテスト支援装置600は、図30に示すフローチャートに従って、暗号プログラムのテストを順次実施し、その結果を蓄積していく。

【0170】まず、開始直後にステップ608でテストスケジュール6010で示される全ての項番のテストが終了したかを調べる。そうであれば、処理を終了する。さもなければステップ609に移る。

【0171】ステップ609では、テストケース選択装置602によって、テストスケジュール格納装置601に格納されている図29に示すようなテストスケジュール6010に従って、暗号プログラムテストケース格納装置305から、次のテストケース304を選択する。

【0172】ステップ610では、プログラム実行装置603によって、前記選択されたテストケース304の構成要素である暗号鍵3042と平文3043とを、暗号プログラム格納装置605に格納されているテスト対象の暗号プログラムに与えて実行させ、暗号文を生成する。

【0173】ステップ611では、比較装置604によって、テスト対象の暗号プログラムの実行結果である暗号文と前記選択されたテストケース304の構成要素である暗号文3044とを比較する。

【0174】ステップ612では、前記選択されたテストケース304の構成要素であるID3041と比較装置604の比較結果とからなるテスト結果607を暗号プログラムテスト結果格納装置606に格納し、ステップ608に戻る。

【0175】このように、第5の実施形態によれば、前述のテストケース生成装置300によって生成された暗号プログラムテストケース304をテストスケジュール601に従ってテスト対象の暗号プログラムに与え、そのテスト対象の暗号プログラムの信頼性をテストすることができる。

【0176】なお、上記の各実施形態で説明した開発支援装置における各機能ブロックは、コンピュータが実行可能なプログラムによって実現することができ、そのプログラムはCDROM等の記憶媒体に格納されてユーザに提供される。

【0177】

【発明の効果】以上説明したように本発明によれば、予め定義したブロック暗号アルゴリズム記法に基づいてブロック暗号アルゴリズムの記述と編集を効率良く支援し、新しい解説技術に耐えられる新しい暗号アルゴリズムの開発を効率良く支援することができる。

【0178】また、ブロック暗号アルゴリズム記法によって記述されたブロック暗号アルゴリズムを直接解釈実行することによって、該ブロック暗号アルゴリズムを実現するプログラムをテストする際に使用できるテストケースを自動的に生成し、暗号プログラムのテストを効率良く支援することができる。

【0179】さらに、ブロック暗号アルゴリズム記法によって記述されたブロック暗号アルゴリズムから、プログラム開発の目的に適したプログラムの設計仕様を自動生成し、暗号プログラムの生成を効率良く支援することができる。

【0180】また、ブロック暗号アルゴリズムから自動生成された前記プログラム設計仕様から、該ブロック暗号アルゴリズムを実現するプログラムを自動生成し、暗号プログラムの生成を効率良く支援することができる。

【0181】さらに、暗号プログラム開発のテスト工程において、ユーザによって指定されたスケジュールに従って、自動生成されたテストケースを用いて、暗号プロ

グラムを効率良くテストすることができるなど、新しい解説技術に耐えられる高信頼性の暗号プログラムを開発するうえでの効率を向上させるのに極めて有効である。

【図面の簡単な説明】

【図1】本発明で用いる暗号アルゴリズム仕様記述記法で使用するダイアグラム表現を示す図である。

【図2】図1と共に暗号アルゴリズム仕様記述記法で使用するダイアグラム表現を示す図である。

【図3】ブロック暗号アルゴリズムDESの全体の機能を記述した例を示すダイアグラム表現図である。

【図4】図3における演算定義の機能を記述した例を示すダイアグラム表現図である。

【図5】図3におけるブロック暗号アルゴリズムDESの鍵生成部の機能を記述した例を示すダイアグラム表現図である。

【図6】図3、図4、図5で使用されている転字、シフト、代入の演算に用いるテーブルの詳細を示す図である。

【図7】本発明の第1の実施形態であるブロック暗号アルゴリズム開発支援装置の機能構成図である。

【図8】図7における暗号ダイアグラム解釈実行装置の機能構成図である。

【図9】図7における暗号ダイアグラム実行環境検査装置の機能構成図である。

【図10】暗号ダイアグラム解釈実行装置2の動作を示すフローチャートである。

【図11】暗号ダイアグラム実行環境格納装置の内容の例を示す図である。

【図12】暗号ダイアグラム実行環境格納装置の内容の例を示す図である。

【図13】暗号ダイアグラム実行環境格納装置の内容の例を示す図である。

【図14】暗号ダイアグラム解釈実行装置の動作例を示す説明図である。

【図15】図14の続きを示すフローチャートである。

【図16】図15の続きを示すフローチャートである。

【図17】本発明の第2の実施形態であるブロック暗号プログラムテストケース生成装置の機能構成図である。

【図18】本発明の第3の実施形態であるブロック暗号プログラム設計仕様生成装置の機能構成図である。

【図19】ダイアグラム設計仕様対応テーブル405の内容の定義例を示す説明図である。

【図20】図19の続きを示す説明図である。

【図21】図3、図4、図5のDESアルゴリズムに対応するプログラム設計仕様の例を示す説明図である。

【図22】図21の続きを示す説明図である。

【図23】本発明の第4の実施形態であるブロック暗号プログラム生成装置の機能構成図である。

【図24】設計仕様プログラム対応テーブル505の内容の定義例を示す説明図である。

【図25】図24の続きを示す説明図である。

【図26】図21、図22のプログラム設計仕様に対応して生成されたプログラムの例を示すプログラムリスト図である。

10 【図27】ビット演算用のC言語ライブラリのヘッダファイルの例を示す図である。

【図28】本発明の第5の実施形態であるブロック暗号プログラムテスト支援装置の機能構成図である。

【図29】テストスケジュールの一例を示す図である。

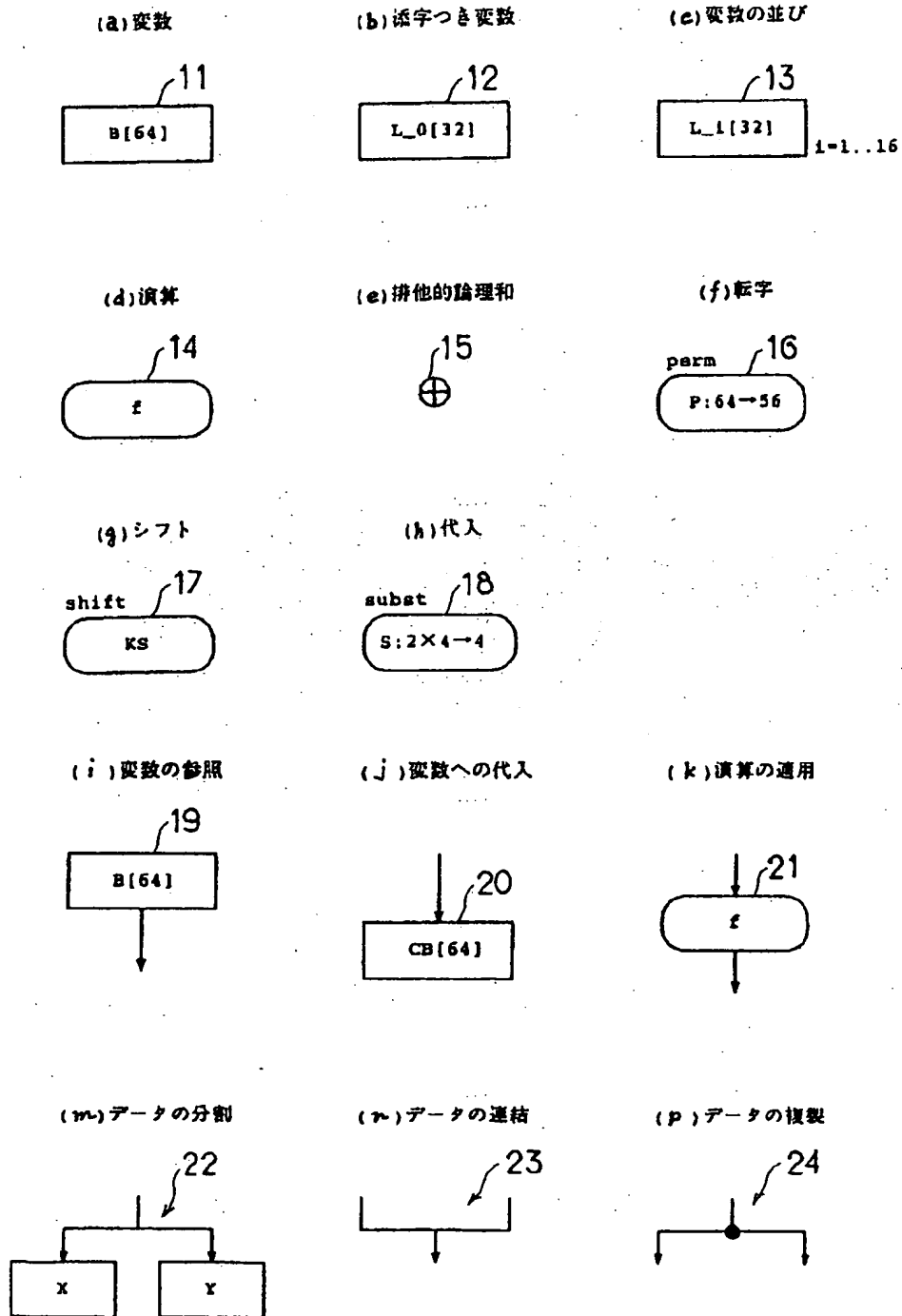
【図30】ブロック暗号プログラムテスト支援装置の動作を示すフローチャートである。

【符号の説明】

200…暗号アルゴリズム開発支援装置、201…暗号ダイアグラム編集装置、202…暗号ダイアグラム解釈
20 実行装置、203…暗号ダイアグラム実行環境検査装置、204…暗号ダイアグラム格納装置、205…暗号ダイアグラム実行環境格納装置、206…ダイアグラム走査装置、207…計算可能性検査装置、208…初期値設定装置、209…値計算装置、210…検査対象指定装置、211…検査結果表示装置、300…暗号プログラムテストケース生成装置、301…ID計数装置、302…鍵生成装置、303…平文生成装置、304…ブロック暗号プログラムのテストケース、305…暗号プログラムテストケース格納装置、400…暗号プログラム設計仕様生成装置、401…ダイアグラム構文解析装置、402…設計仕様生成装置、403…ダイアグラム構文木格納装置、404…暗号プログラム設計仕様書格納装置、405…ダイアグラム設計仕様対応テーブル、500…暗号プログラム生成装置、501…設計仕様構文解析装置、502…プログラム生成装置、503…設計仕様構文木格納装置、504…暗号プログラム格納装置、505…設計仕様プログラム対応テーブル、506…コンパイラ/リンカ、507…ライブラリ、508…実行可能形式暗号プログラム、600…暗号プログラムテスト支援装置、601…テストスケジュール格納装置、602…テストケース選択装置、603…プログラム実行装置、604…比較装置、605…暗号プログラム格納装置、606…暗号プログラムテスト結果格納装置、607…テスト結果。

【図1】

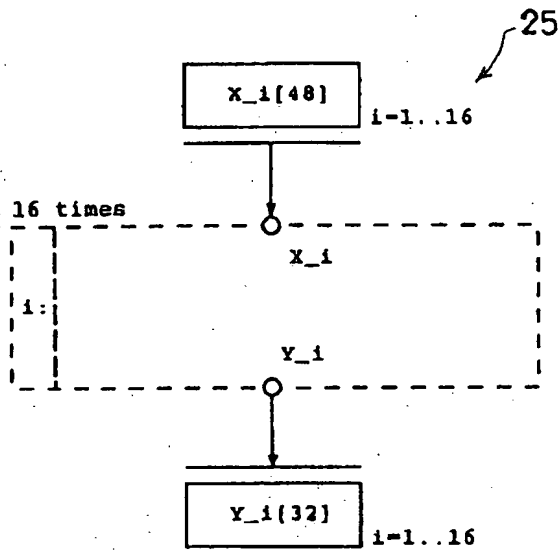
図 1



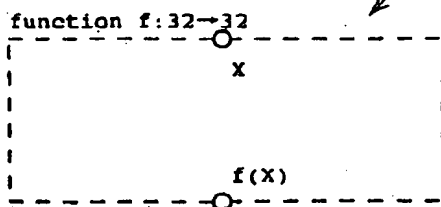
【図2】

図2

(a) 繰り返し



(b) 演算の定義



【図6】

図6

1	2	3	4	5	6	7	8	9	10	11	12	...	62	63	64
58	50	42	34	26	18	10	2	60	52	44	36	...	23	15	7

(a) 転字 IP

1	2	3	4	5	6	7	8	9	10	11	12	...	46	47	48
32	1	2	3	4	5	6	7	8	9	10	11	...	31	32	1

(b) 拡大転字 BP

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1
2
3

(c) 代入 S_1

シフト演算	K_1	K_2	K_3	K_4	...	K_{15}	K_{16}
左シフト数	1	1	2	2	...	2	1

(d) シフト演算

【図11】

図11

(a) DES全体

構成要素	計算可能か?	値
B	NO	-
IP	NO	-
L_0	NO	-
R_0	NO	-
L_1	NO	-
R_1	NO	-
:	NO	-
L_{16}	NO	-
R_{16}	NO	-
K_1	NO	-
:	NO	-
K_{16}	NO	-
繰り返し部 $i=1$	NO	-
:	NO	-
繰り返し部 $i=16$	NO	-
IP^{-1}	NO	-
CB	NO	-

(b) DES全体の繰り返し部 $i=1$

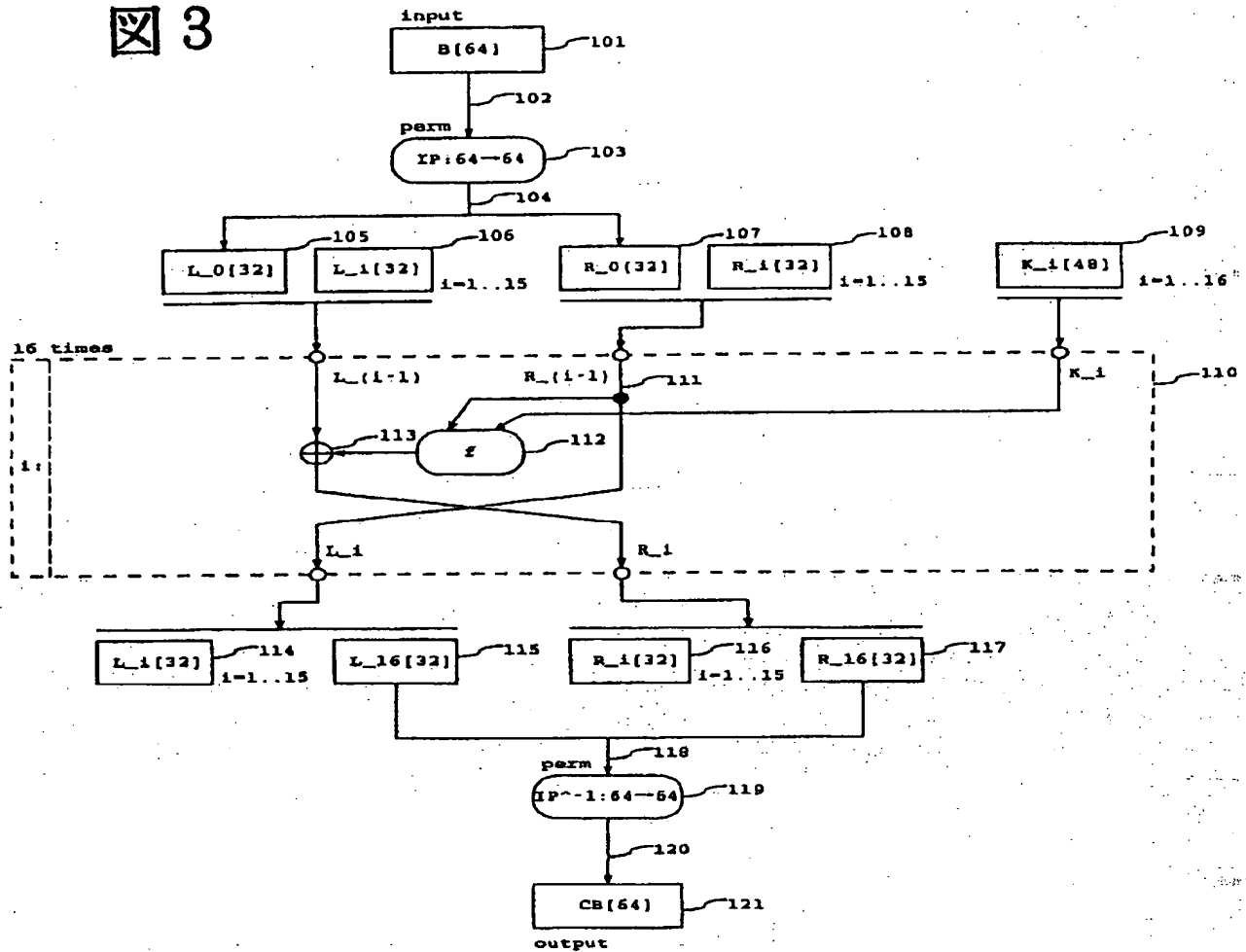
1110

構成要素	計算可能か?	値
L_0	NO	-
R_0	NO	-
K_1	NO	-
f	NO	-
\oplus	NO	-
L_1	NO	-
R_1	NO	-

$i=1..16$

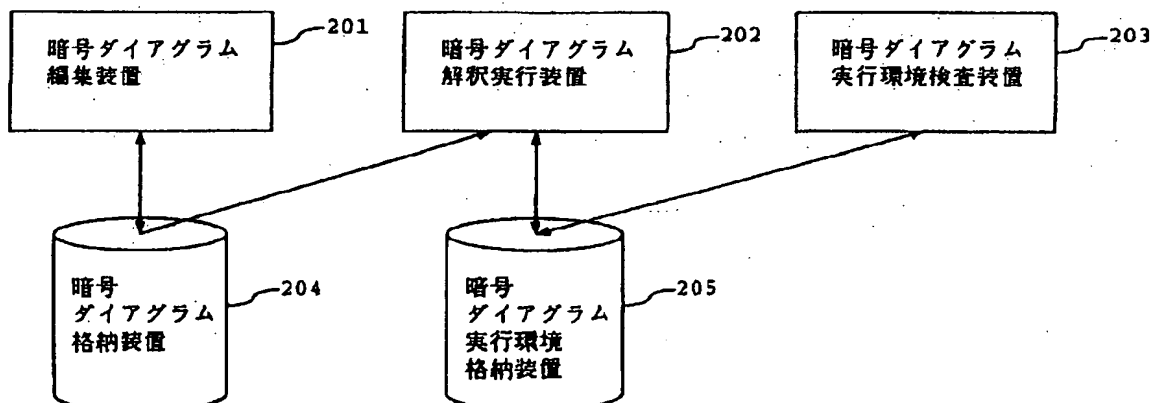
【図3】

図 3



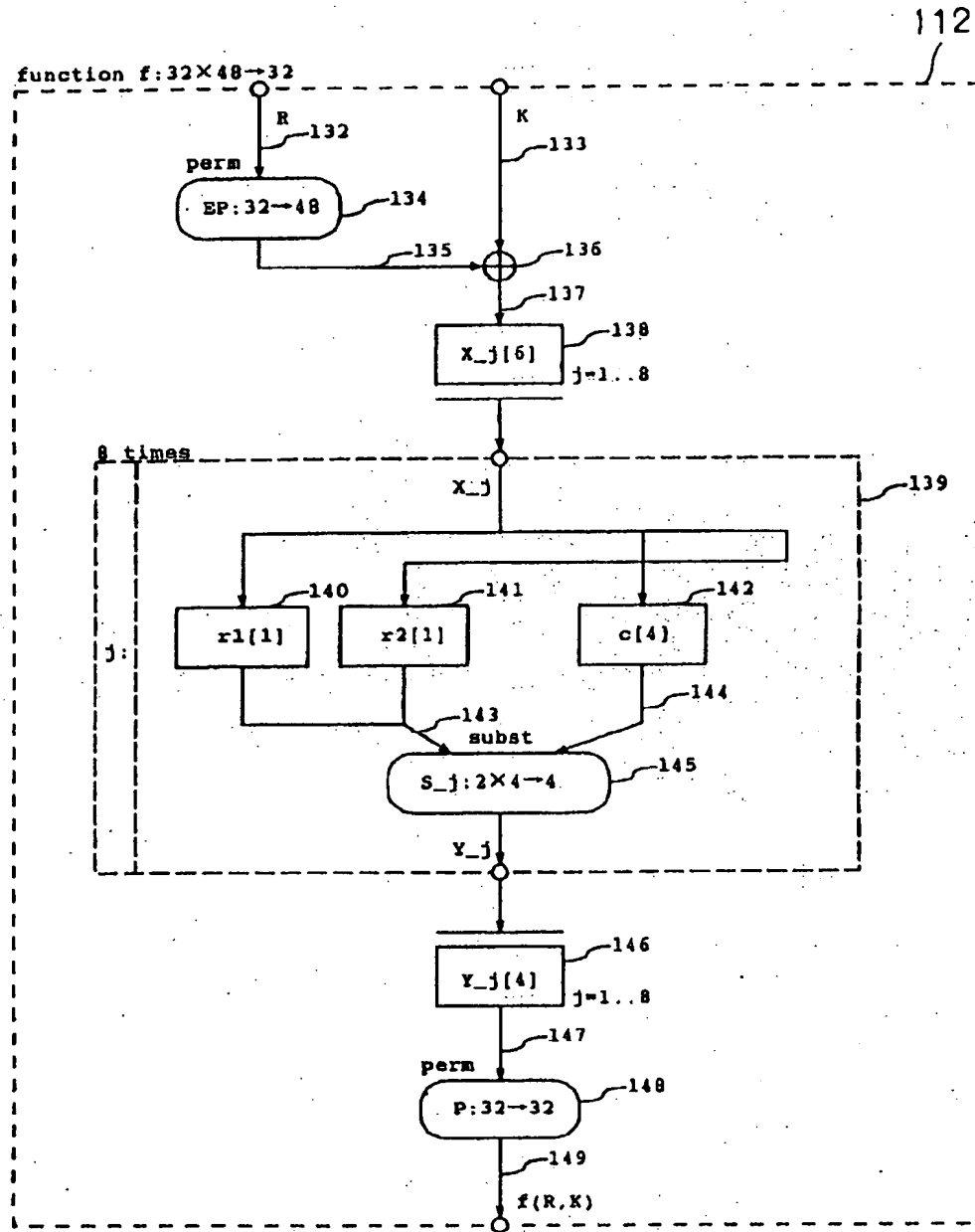
【図7】

図 7



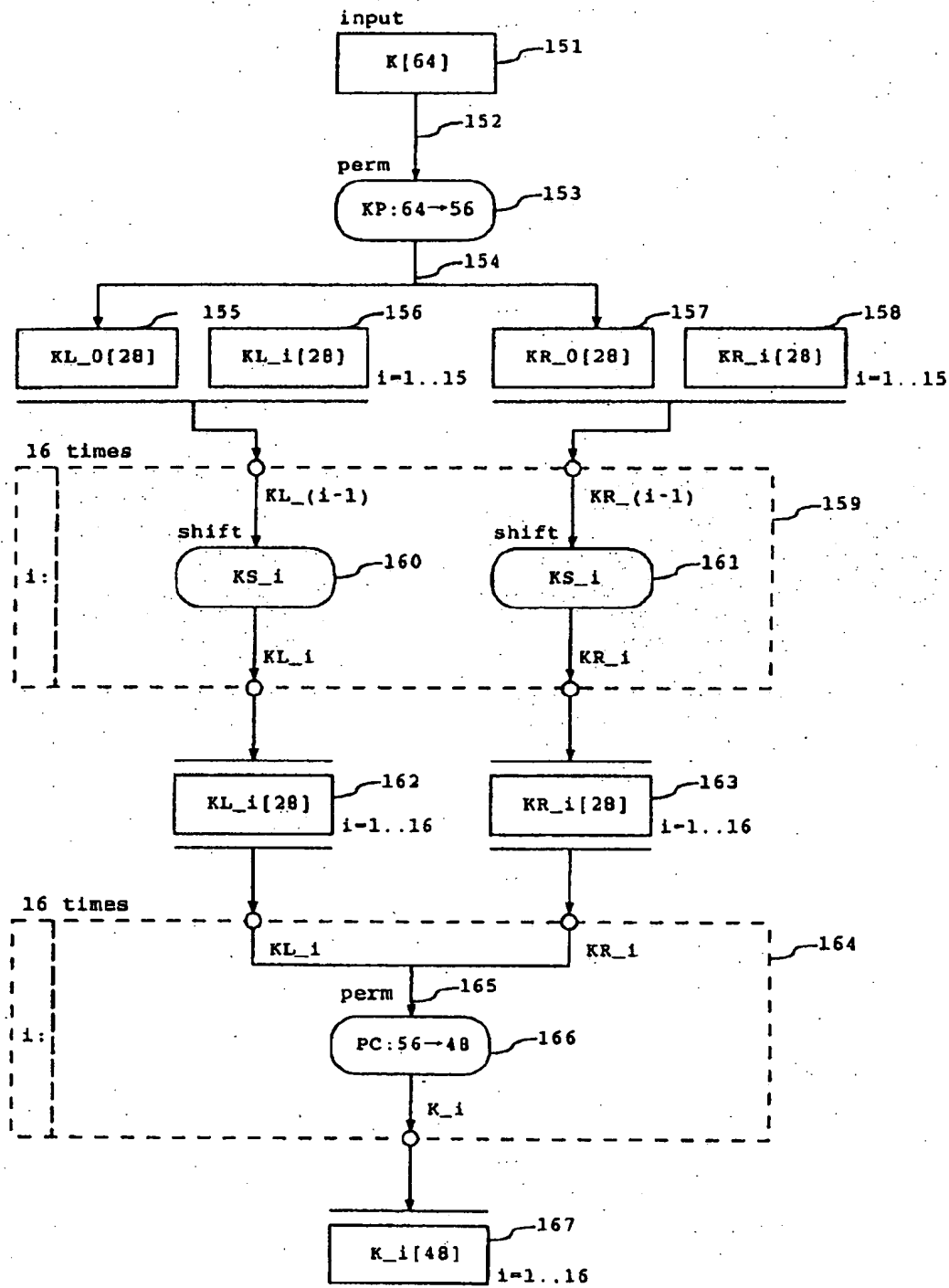
【図4】

図 4



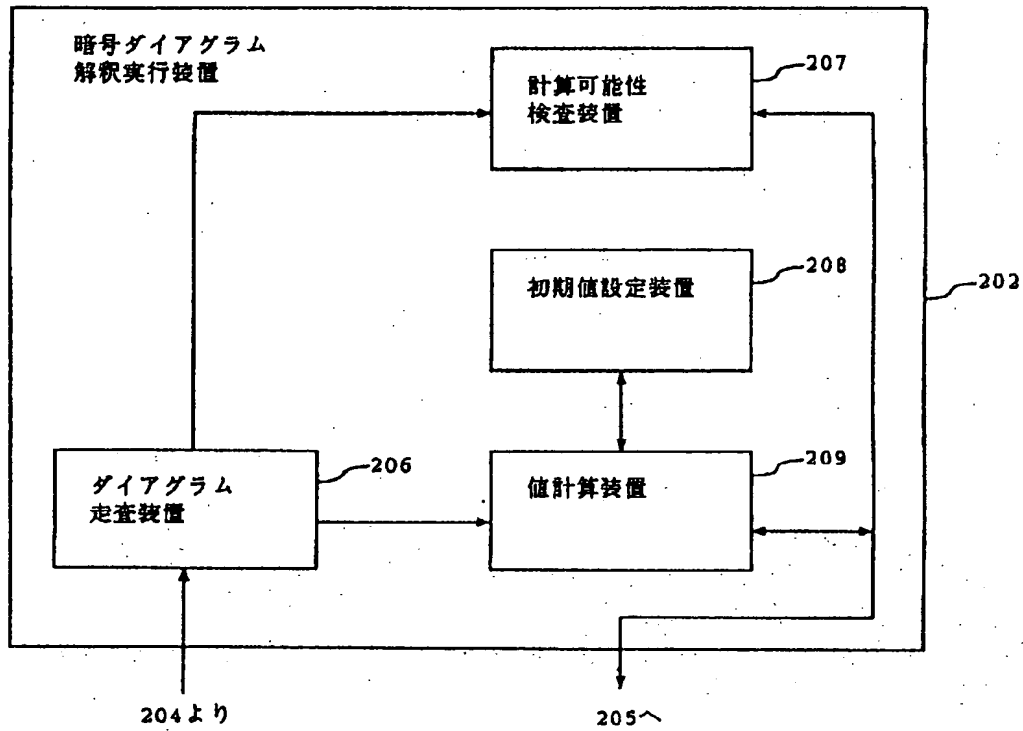
【図5】

図5



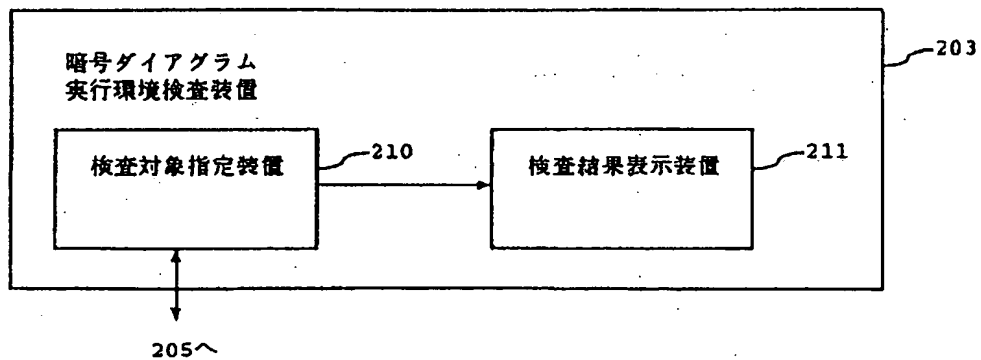
【図8】

図 8



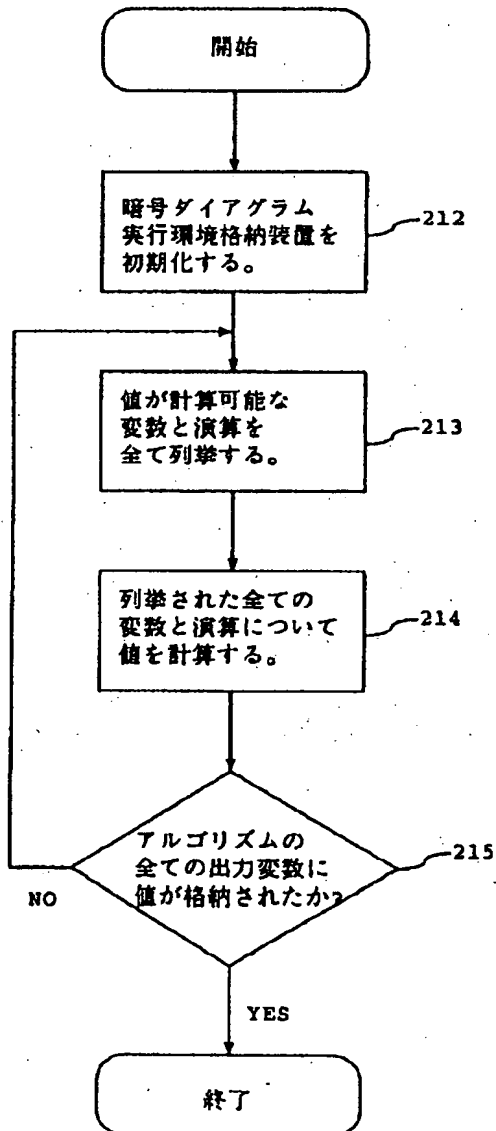
【図9】

図 9



【図 10】

図 10



【図 12】

図 12

生成要素	計算可能か?	値
K	NO	-
KP	NO	-
KL_0	NO	-
KR_0	NO	-
KL_1	NO	-
KR_1	NO	-
:	NO	-
KL_16	NO	-
KR_16	NO	-
繰り返し部(1) i=1	NO	-
:	NO	-
繰り返し部(1) i=1-16	NO	-
繰り返し部(2) i=1	NO	-
:	NO	-
繰り返し部(2) i=1-16	NO	-
X_1	NO	-
:	NO	-
X_16	NO	-

【図 13】

図 13

生成要素	計算可能か?	値
KL_0	NO	-
KR_0	NO	-
KL_1	NO	-
KR_1	NO	-
KL_1	NO	-
KR_1	NO	-

(a) 繰り返し部(1) i=1 1101 1102 1103 1130 i=1-16

生成要素	計算可能か?	値
KL_1	NO	-
KR_1	NO	-
PC	NO	-
X_1	NO	-

(b) 繰り返し部(2) i=1 1140 i=1-16

【 ㊦ 1 4 】

图 14

		1401										1402		
変異要素		開始	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
X	計算可能か? 値	NO -	YES X											
XP	計算可能か? 値	NO -	YES XP (X)											
XL_0	計算可能か? 値	NO -	YES L_0											
XR_0	計算可能か? 値	NO -	YES R_0											
XL_1	計算可能か? 値	NO -	YES L_1											
XR_1	計算可能か? 値	NO -	YES R_1											

【图 15】

图 15

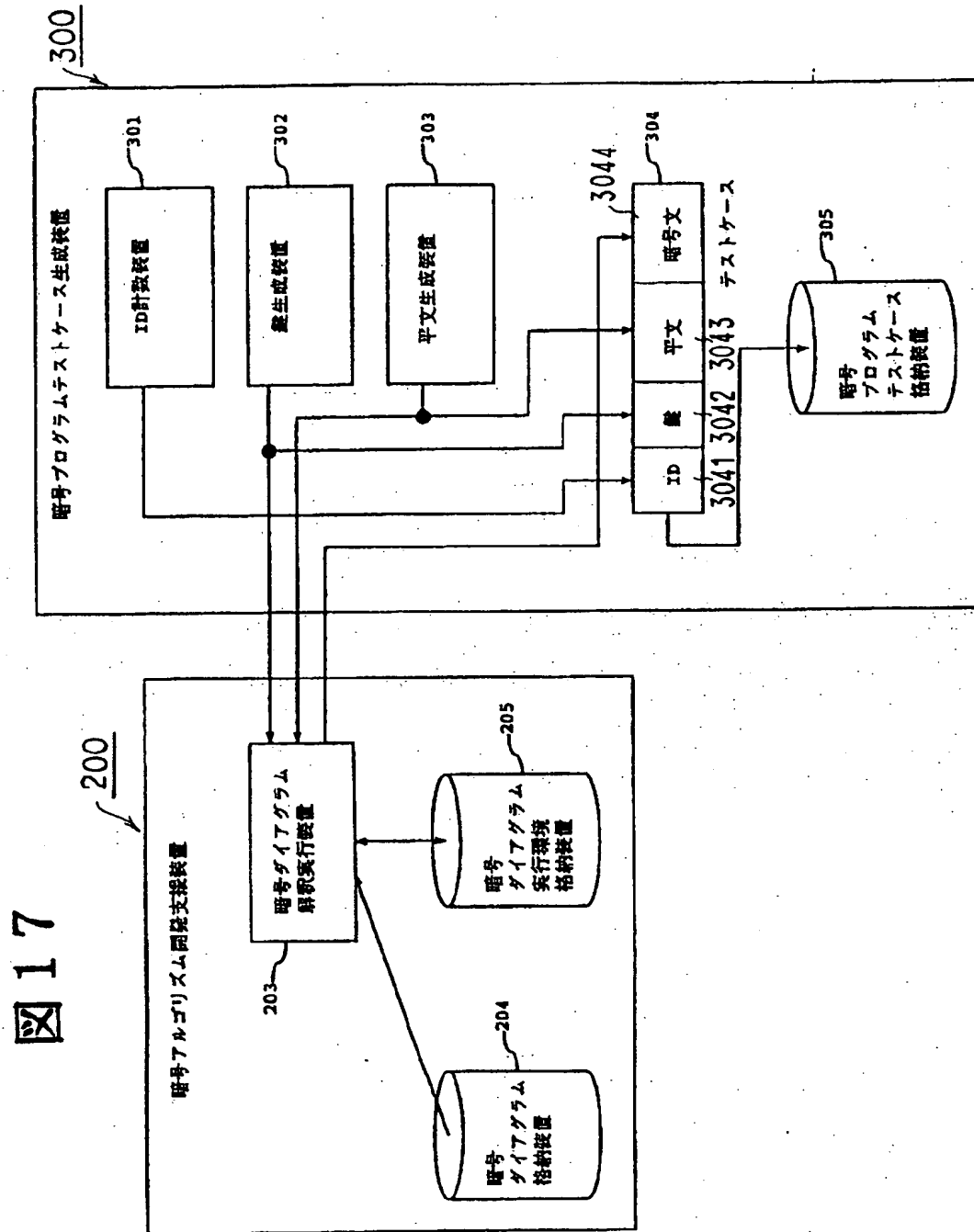
1401			1402											
検査項目		開始	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
繰り返し部(1) 1-1			計算可能か?	NO	YES									
入力	KL_0	計算可能か?	NO	YES										
		値	-	L_0										
	XR_0	計算可能か?	NO	YES										
		値	-	R_0										
	KL_1	計算可能か?	NO	YES										
		値	-	L_1										
	XR_1	計算可能か?	NO	YES										
		値	-	R_1										
出力	KL_1	計算可能か?	NO	YES										
		値	-	L_1										
	XR_1	計算可能か?	NO	YES										
		値	-	R_1										
繰り返し部(1) 1-2			計算可能か?	NO	YES									

【例 16】

图 16

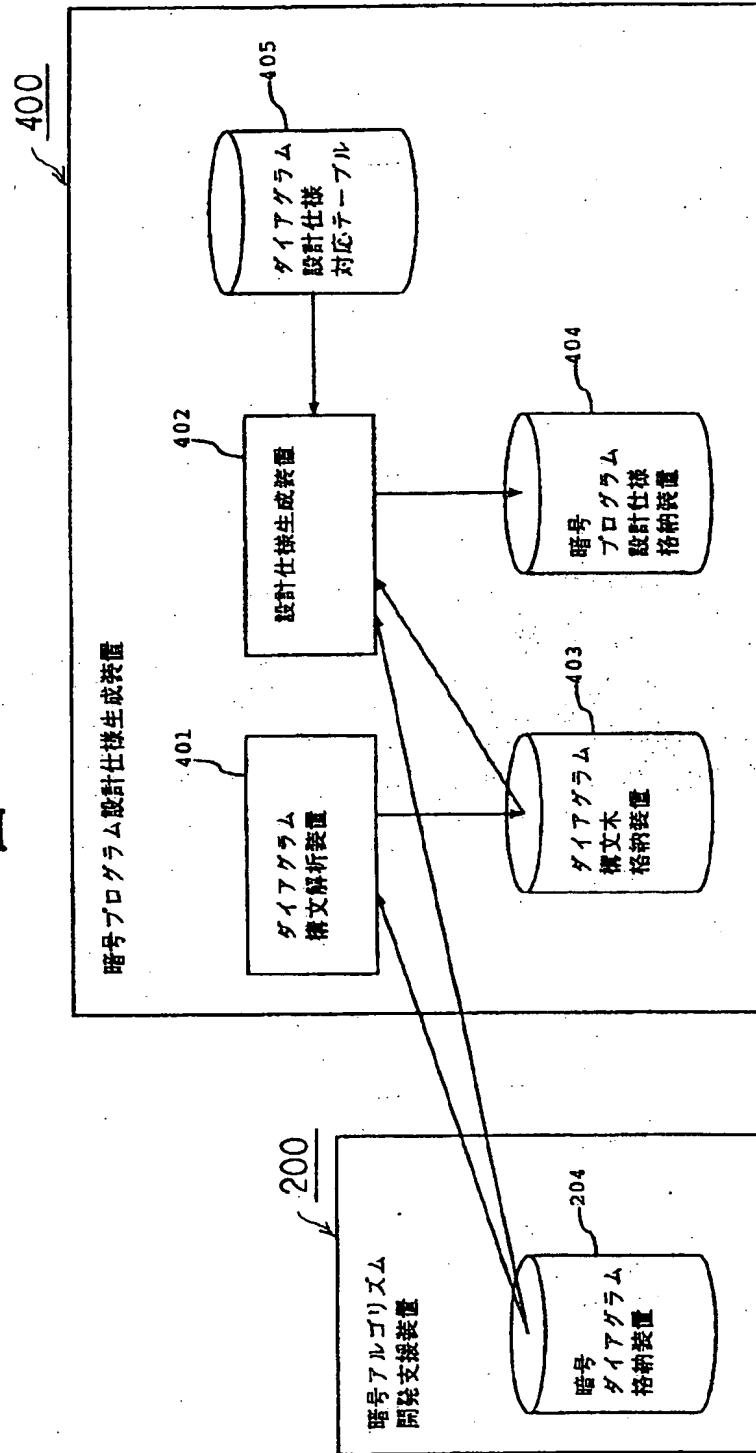
			1401	1402											
管理要素		開始	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	
繰り返し部(3) 1-1		計算可能か?	NO	YES											
入力	IX_1	計算可能か?	NO	YES											
		値	-	L_1											
	IX_1	計算可能か?	NO	YES											
		値	-	R_1											
PC		計算可能か?	NO	YES											
		値	-	PC(L_1, R_1)											
出力	R_1	計算可能か?	NO	YES											
		値	-	PC(L_1, R_1)											
R_1		計算可能か?	NO	YES											
		値	-	PC(L_1, R_1)											

【図17】



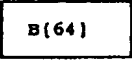
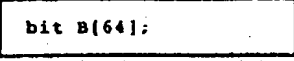
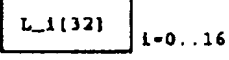
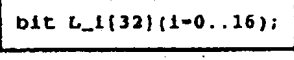
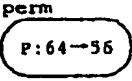
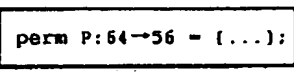
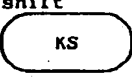
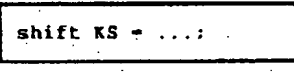
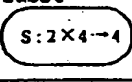
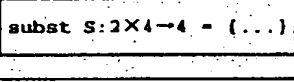
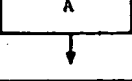
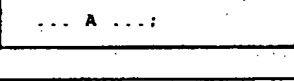
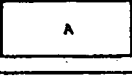
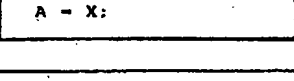
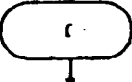
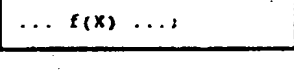
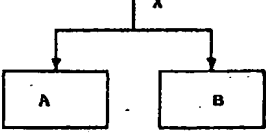
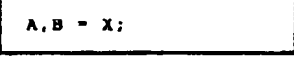

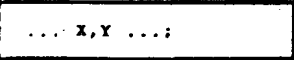
【図18】

図18



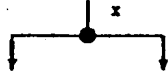
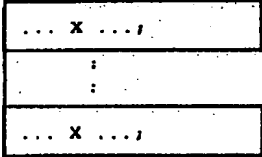
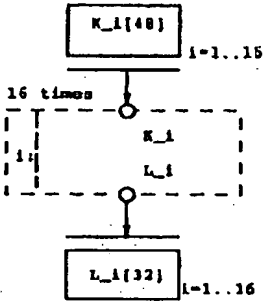

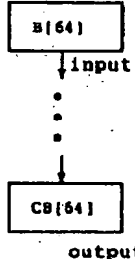
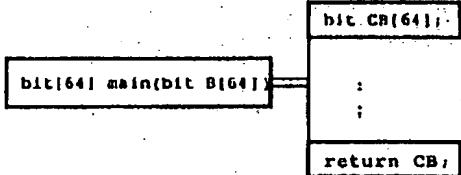
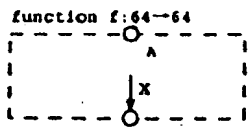
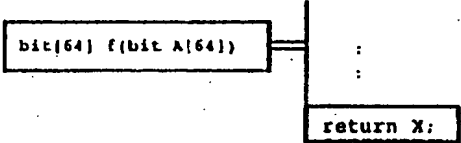
【図19】

図 1 9

項番	項目	4051 シイアグラム表現	4052 設計仕様(PAD)表現
(1)	変数		
(2)	変数の並び		
(3)	転字		
(4)	シフト		
(5)	代人		
(6)	変数の参照		
(7)	変数への代人		
(8)	演算の適用		
(9)	データの分割		
(10)	データの連結		

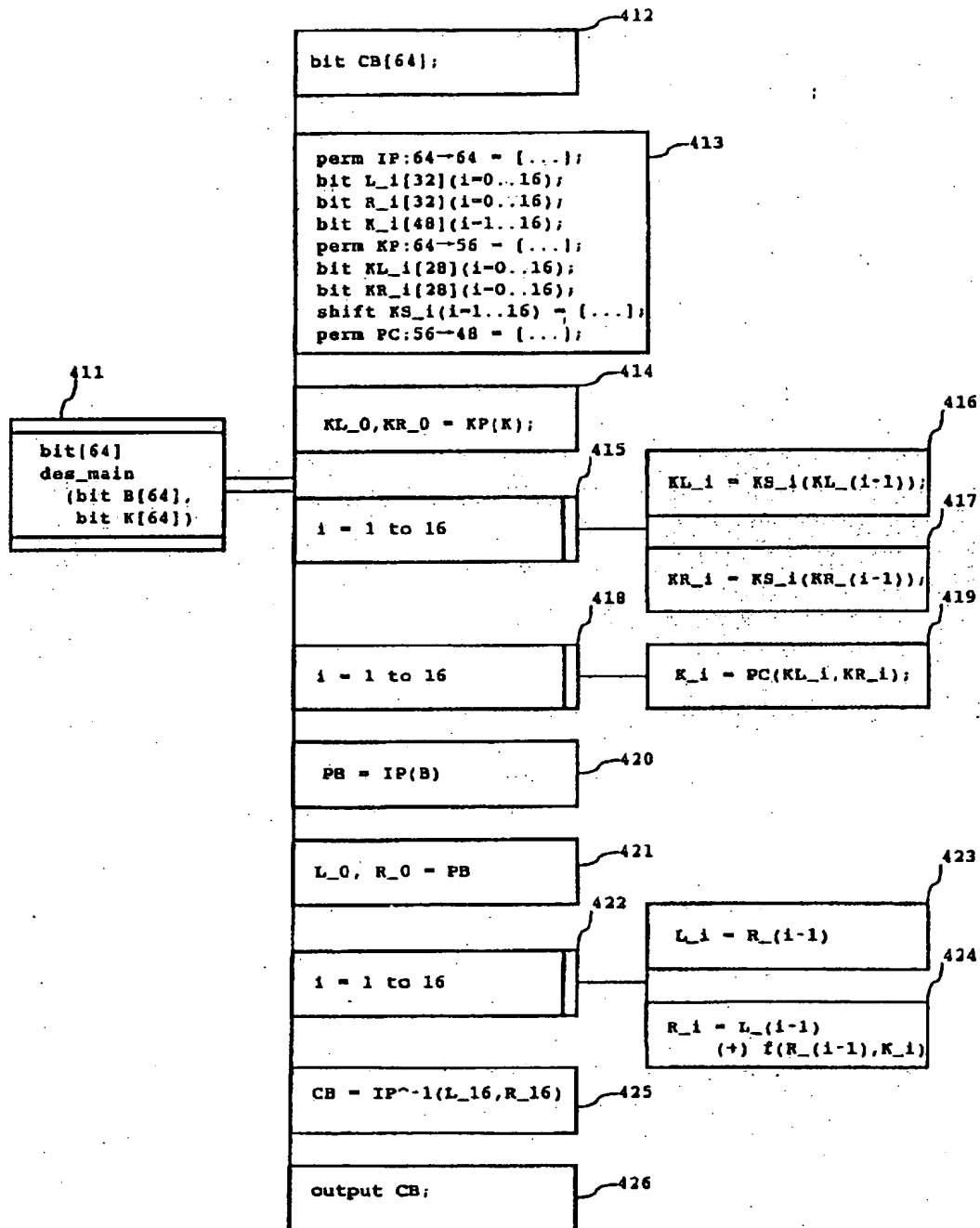
【図 20】

図 20

項番	項目	ダイアグラム表現	設計仕様(PAD)表現
(11)	データの複製		
(12)	繰り返し		
(13)	入出力変数		
(14)	演算の定義		

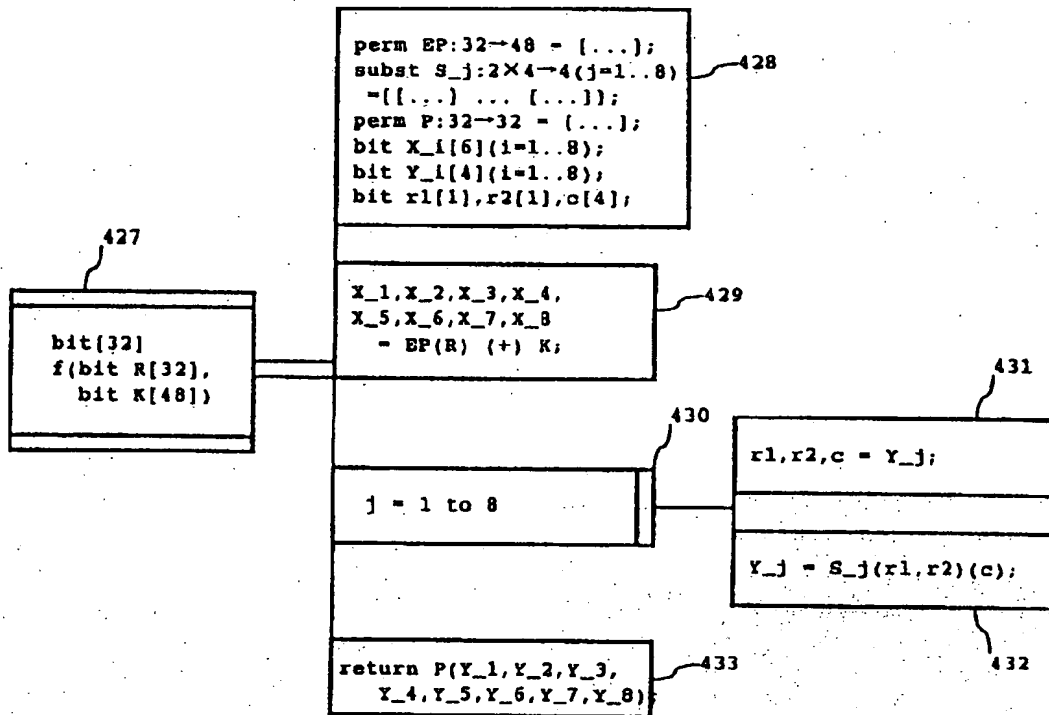
【図21】

図 2 1



【図22】

図22



【図27】

図27

```

/*
 * ビット演算用のC言語ライブラリ
 * long で64ビットデータを表せることを仮定している。
 */

typedef long bit64;
typedef long bit56;
typedef long bit48;
typedef long bit32;
typedef long bit28;
typedef char bit6;
typedef char bit4;
typedef char bit2;
typedef char bit1;

extern long perm(int[], long);
extern long xor(int, long, long);
extern long shift(int, long, int);
extern long concat(int, long, ...);
extern void divide(int, long, long*, int, ...);
extern int* inv(int*, int, int);

```

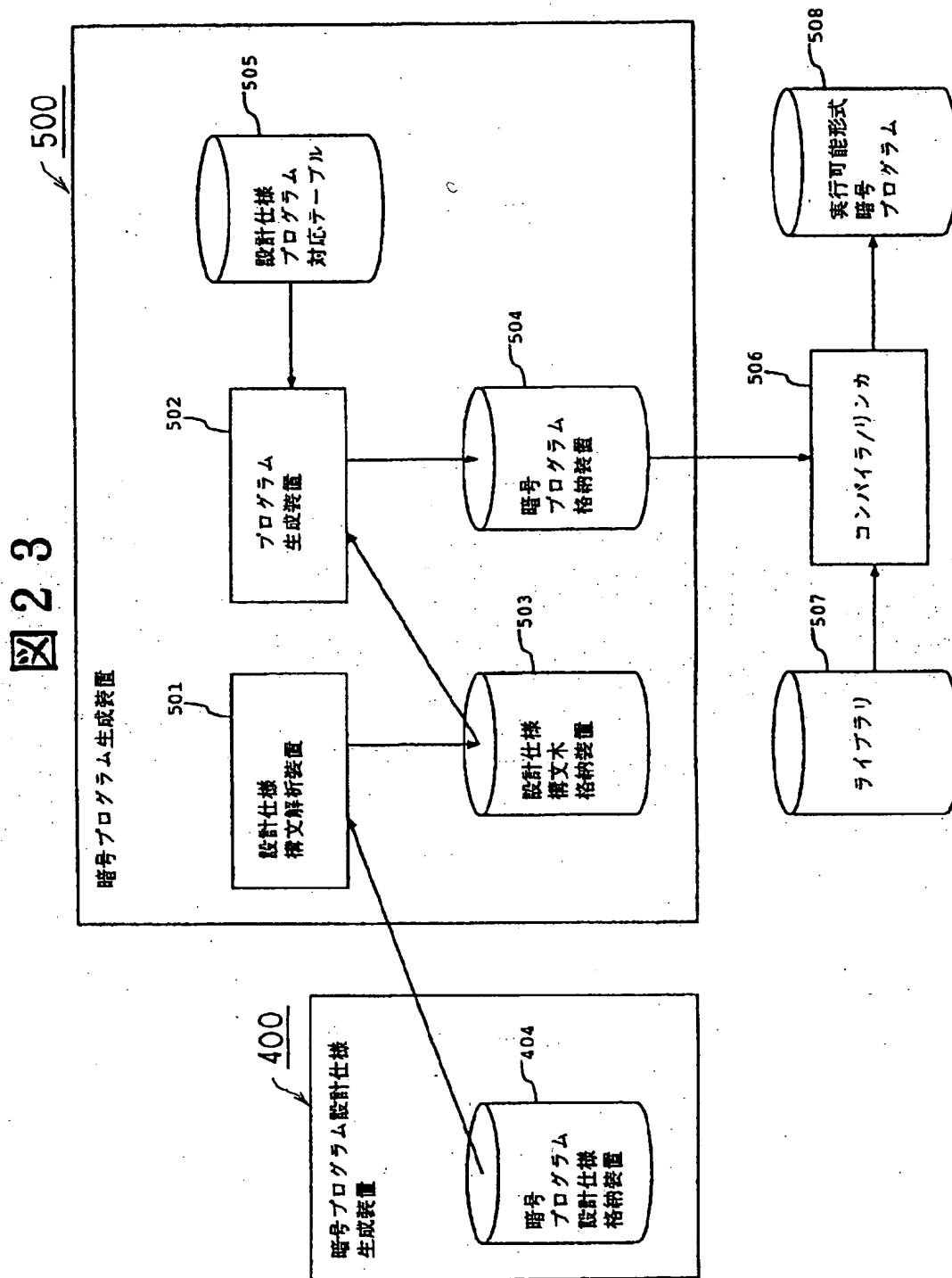
【図29】

図29

6010

項番	テストケースID
1	#010
2	#008
3	#017
4	#024
5	#002

【図23】



【図24】

図 2 4

項番	項目	設計仕様(PAD)表現	C言語表現
(1)	変数宣言	bit B[64];	bit64 B;
(2)	変数並びの宣言	bit L_i[32](1-0..16);	bit32 L[17];
(3)	転字定義	perm P:64→56 = {...};	int P[56] = {...};
(4)	シフト定義	shift KS = ...;	int KS = ...;
(5)	代人定義	subset S:2×4→4 = {...};	bit4 S[4][16] = {...};
(6)	変数の参照(1)	... A ...;	... A ...;
(7)	変数の参照(2)	... A_i ...;	... A[i] ...;
(8)	変数への代人(1)	A = X;	A = X;
(9)	変数への代人(2)	A_i = X;	A[i] = X;
(10)	演算の適用(1)	... f(X) ...;	... f(X) ...;

【図25】

図 2 5

引番	項目	設計仕様(PAD)表現	C言語表現
(11)	演算の適用(2)	5051 	5052 <code>... xor(n,X,Y) ...;</code> X,Yがnビットデータの時
(12)	演算の適用(3)		<code>... perm(P,X) ...;</code> Pがnビットの時
(13)	演算の適用(4)		<code>... perm(inv(P,m,n),X) ...;</code> P:m→nがnビットの時
(14)	演算の適用(5)		<code>... shift(KS,X,n) ...;</code> KSがシフトで Xがnビットデータの時
(15)	演算の適用(6)		<code>... S(r)(c) ...;</code> sが代入の時
(16)	データの分割		<code>divide(X,A,m,B,n);</code> Aがmビットデータ、 Bがnビットデータの時
(17)	データの連結		<code>... concat(X,m,Y,n) ...;</code> Xがmビットデータ、 Yがnビットデータの時
(18)	繰り返し		<code>for (int i=1; i<=16; i++) { L[i] = ... K[i] ...; }</code>
(19)	返値		<code>return X;</code>
(20)	演算定義		<code>extern bit32 f(bit32 A);</code> <code>bit32 f(bit32 A) {</code> ... }

【図26】

図26

(a)

4041

```

extern bit64 des_main(bit64 B, bit64 K);
extern bit32 f(bit32 A, bit48 K);

bit64 des_main(bit64 B, bit64 K) {
    bit64 CS;
    int IP[64] = { ... };
    bit32 L[17], R[17];
    bit48 K[17];
    int KP[56] = { ... };
    bit32 XL[17], XR[17];
    int KS[16] = { ... };
    int PC[48] = { ... };
    divide(parm(KP,K),XL[0],28,XR[0],28);
    for(int i=1; i<=16; i++) {
        XL[i] = shift(XS[i],XL[i-1],28);
        XR[i] = shift(XS[i],XR[i-1],28);
    }
    for(int i=1; i<=16; i++) {
        K[i] = perm(PC,concat(XL[i],28,XR[i],28));
    }
    divide(parm(IP,B),L[0],32,R[0],32);
    for(int i=1; i<=16; i++) {
        L[i] = R[i-1];
        R[i] = xor(32,L[i-1],f(R[i-1],K[i]));
    }
    CS = perm(inv(IP,54),concat(L[16],28,R[16],28));
    return CS;
}

```

(b)

4042

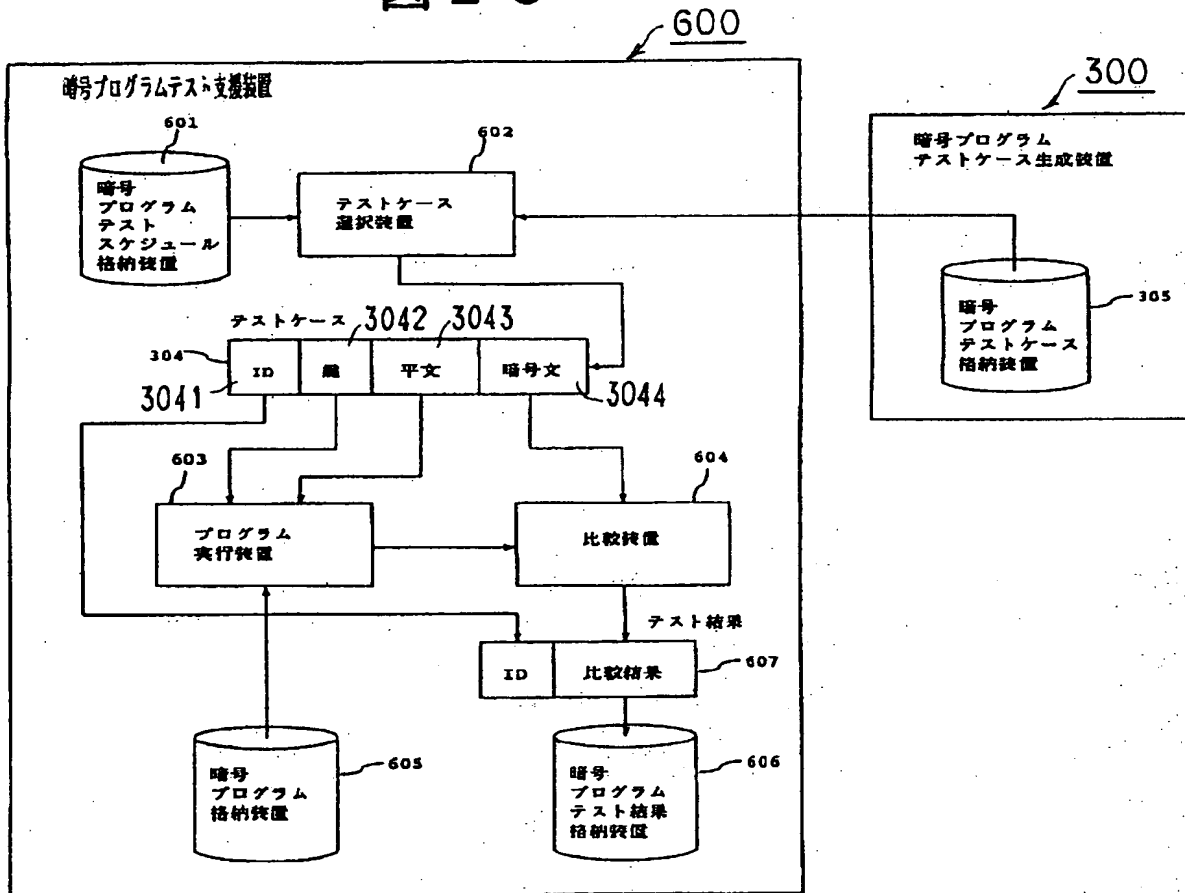
```

bit32 f(bit32 B, bit48 K) {
    int SP[48] = { ... };
    bit4 S[8][4][17] = { ... };
    int P[32] = { ... };
    bit6 X[9];
    bit4 Y[9];
    bit1 x1,x2;
    bit4 o;
    divide(xor(48,perm(SP,B),K),
        X[1],6,X[2],6,X[3],6,X[4],6,X[5],6,X[6],6,X[7],6,X[8],6);
    for(int i=1; i<=8; i++) {
        divide(X[i],x1,1,6,4,x2,1);
        Y[i] = S[i][concat(x1,1,x2,1)][o];
    }
    return perm(P,concat(Y[1],4,X[2],4,X[3],4,X[4],
        4,X[5],4,X[6],4,X[7],4,X[8],4));
}

```

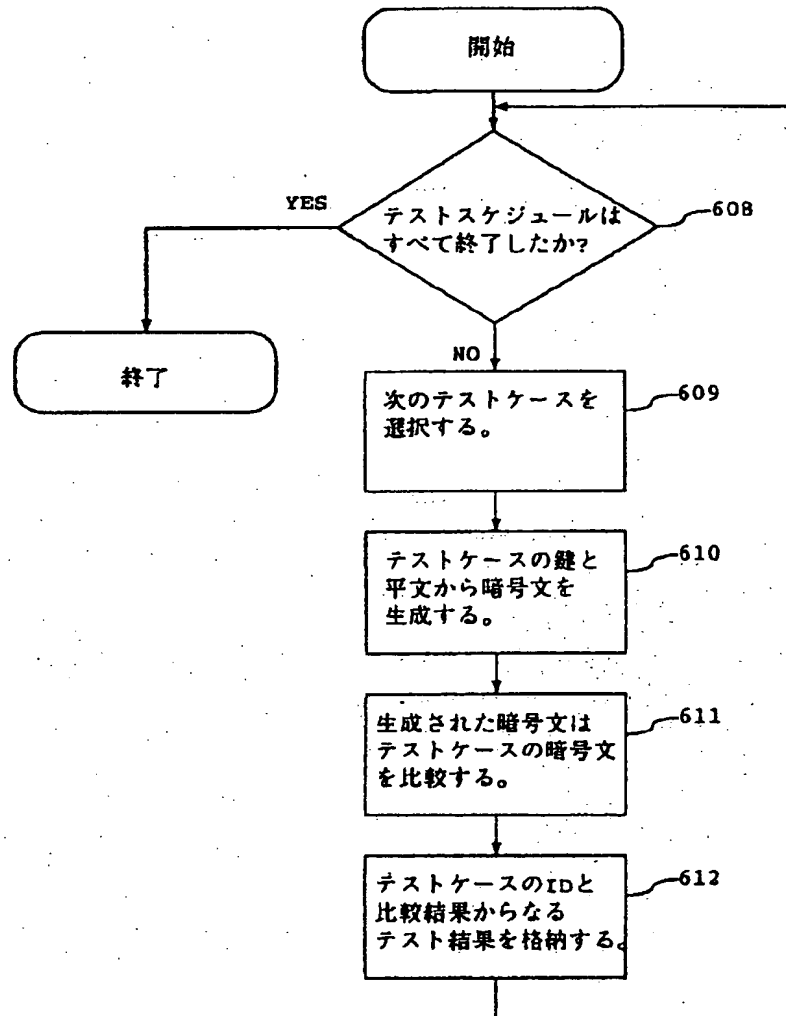
【図28】

図 28



【図30】

図 3 0



フロントページの続き

(51) Int. Cl.⁶H 0 4 L 9/06
9/10

識別記号

F I

H 0 4 L 9/00

6 1 1 Z

6 2 1 Z

(72) 発明者 堤 俊之

神奈川県横浜市中区尾上町6丁目81番地
日立ソフトウェアエンジニアリング株式会
社内